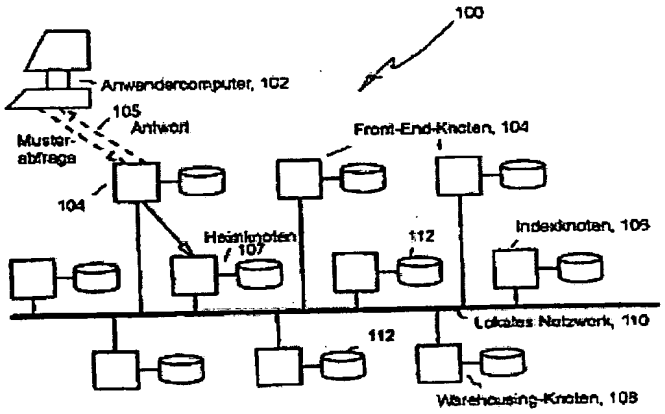
Knowledge extraction method for distributed computer databank system processes received question for retrieval of information corresponding to answer

Patent number:	DE10014757
Publication date:	2001-09-27
Inventor:	BACLAWSKI KENNETH P (US)
Applicant:	JARG CORP (US)
Classification:	
- international:	G06F17/30; G06F17/60
- european:	G06F17/30H; G06F17/30N; G06F17/30T
Application number:	DE20001014757 20000324
Priority number(s):	DE20001014757 20000324

Abstract of **DE10014757**

The knowledge extraction method uses warehouse processing of objects or object positions for extraction of information from a computer databank system, for providing answers to received questions. Object characteristics and object characteristic fragments are located in an index databank and evaluated for identifying a number of sub-questions, with recursive evaluation of the latter and collection of the evaluation results for forming an overall evaluation of the question. Also included are Independent claims for the following: (a) a distributed computer databank system; (b) a method for processing a question for information retrieval from a databank; (c) a computer program product for processing a question for information retrieval from a databank system



(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENT- UND MARKENAMT

(12) **Offenlegungsschrift**

(10) **DE 100 14 757 A 1**

(51) Int. Cl.[7]:
G 06 F 17/30
G 06 F 17/60

(21) Aktenzeichen: 100 14 757.7
(22) Anmeldetag: 24. 3. 2000
(43) Offenlegungstag: 27. 9. 2001

DE 100 14 757 A 1

(71) Anmelder:
Jarg Corp., Waltham, Mass., US

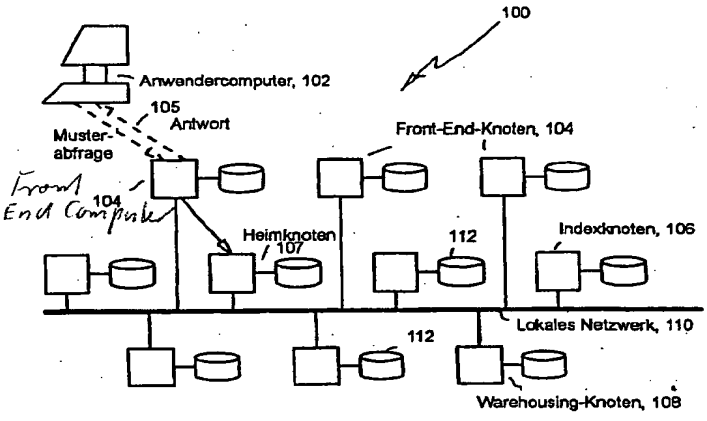
(74) Vertreter:
Kindermann, M., Pat.-Anw., 71032 Böblingen

(72) Erfinder:
Baclawski, Kenneth P., Waltham, Mass., US

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) System und Verfahren zur Wissensextraktion

(57) Eine Informationswiedergewinnungseinrichtung für die Verarbeitung einer Abfrage zur Wiedergewinnung von Informationen aus einer Datenbank besitzt einen Mechanismus zum Auffinden einer Anzahl an Merkmalen und Merkmalsfragmenten in einer Indexdatenbank; einen Evaluierungsmechanismus zur Identifizierung einer Anzahl an Unterabfragen einer Anzahl an Ebenen, die in der Abfrage enthalten sind, und zur rekursiven Evaluierung der Unterabfrage unter Verwendung der einzelnen gefundenen Merkmale und Merkmalsfragmente; und einen Mechanismus zum Sammeln und Speichern einer Anzahl an Ergebnissen der rekursiven Evaluierung der Abfrage und der Unterabfrage nach der Berechnung eines Gesamtergebnisses der Abfrage. Ein solches System kann den Bedarf an herkömmlichen Wiedergewinnungssystemen für die Schaffung neuer, separater, zentralisierter Repliken innerhalb des Data Warehouse der Daten, die in den verschiedenen externen Datenbanken gespeichert sind, beseitigen. Die Erfindung kann somit die Probleme der Replizierung solcher Daten in herkömmlichen Systemen vermeiden, in denen die Daten veraltet sein können oder Fehlern unterliegen können, die während der Replazierung für die Datenlagerhaltung entstehen. Stattdessen kann das Data Warehouse eine Indexdatenbank enthalten, die Einträge speichert, welche Daten hinsichtlich der in den externen Datenbanken gespeicherten Informationen zur Verfügung stellen, wie zum Beispiel Informationspositionsbezeichner für diese Daten innerhalb jener ...



DE 100 14 757 A 1

Beschreibung

VERWANDTE ANMELDUNGEN

Diese Anmeldung ist verwandt mit und beansprucht Priorität über die gleichzeitig anhängige, gemeinsam abgetretene, noch unvollständige US-Patentanmeldung Nr. 60/094,350, angemeldet am 28. Juli 1998 von Kenneth P. Baclawski, mit dem Titel "Knowledge Extraction System and Method"; und der Nr. 60/094,110, angemeldet am 24. Juli 1998 von Kenneth P. Baclawski, mit dem Titel "Distributed Object Search System and Method"; deren Offenbarung hiermit als Referenz aufgenommen wird. Diese Anmeldung betrifft auch die ebenfalls anhängige, gemeinsam abgetretene US-Anmeldung Nr. xxx,xxx, angemeldet am selben Tag wie diese, von Kenneth P. Baclawski, mit dem Titel "Distributed Computer Database System And Method For Performing Object Search", deren Offenbarung hiermit als Referenz aufgenommen wird.

BEREICH DER ERFINDUNG

Die Erfindung betrifft ein Computer-Datenbanksystem, und insbesondere verteilte Computer-Datenbanksysteme.

HINTERGRUND DER ERFINDUNG

Organisationen sammeln routinemäßig große Datenmengen über ihre Kunden, Produkte, Arbeitsabläufe und Geschäftsaktivitäten. Die in diesen Daten enthaltenen Erkenntnisse können wichtige Hilfen zum Marketing, zur Verringerung der Betriebskosten sowie für strategische Entscheidungsfindungen darstellen. Wenn es zum Beispiel eine starke Beziehung zwischen den Kunden, die ein Produkt kaufen möchten, und jenen Kunden, die ein anderes Produkt kaufen möchten, gibt, dann besteht die Wahrscheinlichkeit, daß jene Kunden, die dieses Produkt gekauft haben, auch Interesse am Kauf des anderen Produkts haben können.

Die analytische Verarbeitung von Daten erfolgt primär unter Verwendung statistischer Methoden zum Extrahieren von Korrelationen und anderen Mustern in den Daten. Diese Art der Verarbeitung wird unter anderem als "Data Mining" (Datenerforschung), Wissenserkundung und Wissensextraktion bezeichnet. Eine Suche nach einem spezifischen Muster oder einer Art von Muster in einer großen Sammlung von Daten wird als Musterabfrage bezeichnet.

Große Unternehmen besitzen und verwalten typischerweise Datenbanken, von denen es sich bei vielen um Transaktionsdatenbanken handelt. Die Anforderungen dieser Datenbanken stehen oftmals in Konflikt mit den Anforderungen des "Data Minings". Transaktionsdatenbanken werden in Echtzeit durch kleine Transaktionen aktualisiert. Beim Data Mining hingegen werden große Musterabfragen verwendet, die nicht in Echtzeit stattfinden müssen. Um diesen Konflikt zu lösen, wird nun allgemein so vorgegangen, daß Daten aus unterschiedlichen Quellen in eine zentralisierte Ressource geladen werden, die man als Data Warehouse (Datenlagerhaus) bezeichnet.

Das Herunterladen und Zentralisieren der Daten aus unterschiedlichen, oft getrennten Quellen erfordert die Durchführung zahlreicher Aufgaben. Die Daten müssen aus den Quellen extrahiert werden; sie müssen in ein gemeinsames, integriertes Datenmodell umgewandelt werden; sie müssen, um fehlerhafte oder falsche Daten zu beseitigen oder zu korrigieren, gereinigt werden, und schließlich müssen sie im zentralen Warehouse integriert und zu einer neuen Datenbank zusammengefaßt werden, in der alle Daten gespeichert sind. Darüber hinaus muß sichergestellt werden, daß sämtliche Vorkommen jeder Geschäftseinheit, wie zum Beispiel Kunde, Produkt oder Mitarbeiter, korrekt identifiziert wurden. Dieses Problem ist als referentielle Integrität bekannt. All dies sind schwierige Aufgaben, besonders jedoch das Sicherstellen der referentiellen Integrität, wenn die von den Datenbanken heruntergeladenen Daten die Geschäftseinheiten geringfügig anders identifizieren. Bei der Technologie des Standes der Technik werden Daten in einer vom Data Mining unabhängigen Aktivität in das Data Warehouse geladen. Im Gegensatz zum Data Mining, für welches es eine umfangreiche Forschungsliteratur und viele kommerzielle Produkte gibt, besitzt das Data Warehousing keine starke theoretische Grundlage und nur wenige gute kommerzielle Produkte.

Da Data Warehouses viele unterschiedliche Datenquellen integrieren, ist es notwendig, ein integriertes Datenmodell für das Data Warehouse sowie eine Datenabbildung zu schaffen, welches Daten von den einzelnen Datenquellen extrahiert, umwandelt und reinigt. Es ist im Stand der Technik bekannt, daß sich reichere Datenmodelle, wie zum Beispiel objektorientierte Datenmodelle, besser für die Festlegung eines solchen integrierten Datenmodells und für die Definierung der Datenabbildung eignen als begrenzte Datenmodelle, wie zum Beispiel das relationale Modell. Dennoch verwenden die meisten Data Warehouses (Datenlagerhäuser) noch immer eine flache Eintragsstruktur, wie zum Beispiel das relationale Modell. Relationale Datenbanken besitzen eine sehr limitierte Datenstruktur, so daß die Erzeugung komplexerer Datenstrukturen mühevoll und fehlerträchtig ist. Einige der Arten von Daten, die für eine Speicherung in einer relationen Datenbank nur sehr schlecht geeignet sind, wären: Textdaten im allgemeinen, Hypertextdokumente im besonderen, Bilder, Töne, Multimediaobjekte und Attribute mit mehreren Werten. Relationale Datenbanken sind auch schlecht für die Darstellung von Dateneinträgen geeignet, die eine sehr große Anzahl an möglichen Attributen besitzen, von denen nur wenige von einem Dateneintrag verwendet werden.

Eine Objektdatenbank besteht typischerweise aus einer Sammlung von Daten oder Informationsobjekten. Jedes Informationsobjekt wird auf einzigartige Weise durch einen Objektbezeichner (OID) gekennzeichnet. Jedes Informationsobjekt kann Merkmale besitzen, und manche Merkmale können zugeordnete Werte besitzen. Informationsobjekte können auch andere Informationsobjekte enthalten oder auf diese verweisen.

Um das Auffinden von Informationen in einer Datenbank, einschließlich einer Data Warehouse-Datenbank, zu unterstützen, werden spezielle Suchstrukturen verwendet, die man als Indexe bezeichnet. Große Datenbanken erfordern entsprechend große Indexstrukturen, um Zeiger zu den gespeicherten Daten zu setzen und zu verwalten. Eine solche Indexstruktur kann größer sein als die Datenbank selbst. Die Technologie des Standes der Technik erfordert einen separaten Index für jedes Attribut oder Merkmal. Diese Technologie kann erweitert werden, um eine Indizierung einer kleinen Anzahl von Attributen oder Merkmalen in einer einzelnen Indexstruktur zu ermöglichen, doch funktioniert diese Technologie nicht sehr gut, wenn es Hunderte oder Tausende von Attributen gibt. Darüber hinaus gibt es einen beträchtlichen Zusatzaufwand im Zusammenhang mit der Wartung einer Indexstruktur. Dies schränkt die Anzahl der Attribute oder Merkmale, die indiziert werden können, ein, so daß jene, die unterstützt werden, sehr sorgfältig ausgewählt werden müssen. Für Transaktionsdatenbanken ist für gewöhnlich ein gutes Verständnis der damit verbundenen Arbeitslast vorhanden, so daß es möglich ist, die Indexe so zu wählen, daß die Leistung der Datenbank optimiert wird.

Für ein Data Warehouse gibt es jedoch normalerweise keine gut definierte Arbeitslast, so daß es viel schwieriger ist, die zu indizierenden Attribute auszuwählen.

Weitere Informationen bezüglich der zuvor genannten Konzepte können den folgenden Publikationen entnommen werden:

1 L. Aiello, J. Doyle, und S. Shapiro, Herausgeber. Proc. Fifth Intern. Conf. an Principles of Knowledge Representation and Reasoning. Morgan Kaufman Publishers, San Mateo, CA, 1996.

2 K. Baclawski, Distributed computer database system and method, Dezember 1997. US-Patent Nr. 5,694,593. Abgetreten an Northeastern University, Boston, MA.

3 A. Del Bimbo, Herausgeber. The Ninth International Conference an Image Analysis and Processing, Band 1311. Springer, September 1997.

4 N. Fridman Noy. Knowledge Representation for Intelligent Information Retrieval in Experimental Sciences. Doktorarbeit, College of Computer Science, Northeastern University, Boston, MA, 1997.

5 M. Hurwicz. Take your data to the cleaners. Byte Magazine, Januar 1997.

6 Y. Ohta. Knowledge-Based Interpretation of Outdoor Natural Color Scenes. Pitman, Boston, MA, 1985.

7 A. Tversky. Features of similarity. Psychological review, 84(4):327–352, Juli 1977.

8 S. Weiss and N. Indurkhya. Predictive Data Mining: A Practical Guide. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1998.

9 J.-L. Weldon and A. Joch. Data warehouse building blocks. Byte Magazine, Januar 1997.

Die Offenbarungen der im Abschnitt "Hintergrund der Erfindung" erwähnten Veröffentlichungen werden hiermit als Referenz aufgenommen.

Es wäre wünschenswert, verbesserte Systeme für das Data Warehousing und das Data Mining zu schaffen, welche viele der leistungsbezogenen und anderen Probleme und Einschränkungen der Systeme des Standes der Technik beseitigen.

ZUSAMMENFASSUNG DER ERFINDUNG

Die vorliegende Erfindung kombiniert die zwei Aktivitäten des Data Warehousings und des Data Minings, wodurch die Grundlage und Unterstützung für das Data Warehousing verbessert werden. Der Begriff Wissensextraktion wird im folgenden für die Integration des Data Warehousing und der Data Mining-Aktivitäten verwendet.

Die Erfindung beruht auf einem System und einem Verfahren zur Verarbeitung einer Abfrage von einem Anwender, einschließlich zum Beispiel einer Abfrage für die Wiedergewinnung von Informationen aus dem Data Warehouse. Das System umfaßt einen Mechanismus zum Finden einer Anzahl an Merkmalen und Merkmalsfragmenten in einer Indexdatenbank; einen Evaluierungsmechanismus zur Identifizierung einer Anzahl an Unterabfragen einer Anzahl an Ebenen, die in der Abfrage enthalten sind, und zur rekursiven Evaluierung der Unterabfragen mit Hilfe der einzelnen gefundenen Merkmale und Merkmalsfragmente; und einen Mechanismus zum Sammeln und Speichern einer Anzahl an Ergebnissen der rekursiven Evaluierung der Abfrage und der Unterabfragen nach dem Berechnen des Gesamtergebnisses der Abfrage.

Mit dem hierin verwendeten Begriff "Evaluierung" wird ein Prozeß bezeichnet, durch den eine Antwort auf eine Abfrage erzeugt wird, gekennzeichnet durch die Wiedergewinnung von Informationen, Informationspositionsbezeichnern oder Daten, welche die Informationen betreffen, und welche den in der Abfrage angegebenen Kriterien entsprechen. Bei der rekursiven Evaluierung handelt es sich um einen Typ der Abfrageevaluierung, bei dem neue Abfragen, sogenannte Unterabfragen, von der Abfrage erzeugt und evaluiert werden. Die solcherart erzeugten Unterabfragen können als Knoten in einem Abfragebaum betrachtet werden, dessen ursprüngliche Abfrage der Basisknoten ist, und wobei jede Unterabfrage eine entsprechende Ebene innerhalb des Baumes einnimmt, die von ihrer Beziehung zu den vorhergehenden Abfragen, von denen sie erzeugt wurde, bestimmt wird. Alle Unterabfragen, das heißt die Vorgängerabfragen und Tochterabfragen, werden rekursiv evaluiert, und die Ergebnisse werden gesammelt, gespeichert und dem Anwender als Antwort auf die Abfrage präsentiert.

Die Erfindung kann den Bedarf an herkömmlichen Wiedergewinnungssystemen für die Schaffung neuer, separater, zentralisierter Repliken innerhalb des Data Warehouse der Daten in den verschiedenen externen Datenbanken beseitigen. Die Erfindung kann somit die Probleme der Replizierung solcher Daten in herkömmlichen Systemen vermeiden, in denen die Daten veraltet oder Fehlern unterliegen können, die während der Replizierung für die Datenlagerhaltung entstehen. Stattdessen kann das Data Warehouse eine Indexdatenbank enthalten, die Einträge speichert, welche Daten hinsichtlich der in den externen Datenbanken gespeicherten Informationen zur Verfügung stellen, wie zum Beispiel Informationspositionsbezeichner für diese Daten innerhalb jener Datenbanken, relationale Informationen und Statistiken. Die Erfindung kann auch ein robustes, vielseitiges Indiziersystem schaffen. Der Index der Erfindung unterstützt zum Beispiel das Indizieren von kärglichen Einträgen, die eine große Anzahl an potentiellen Attributen besitzen, von denen aber nur einige wenige in einem bestimmten Dateneintrag verwendet werden. Die vorliegende Erfindung unterstützt zum Beispiel auch das Indizieren einer sehr großen Anzahl an Attributen in einer im wesentlichen einheitlichen Datenstruktur, wodurch es viel einfacher wird, die zur Erzielung einer hohen Leistung erforderlichen Arbeitslastmerkmale zu bestimmen.

Insbesondere umfaßt das verteilte Computer-Datenbanksystem gemäß eines Aspektes der Erfindung einen oder mehrere Front-End-Computer und einen oder mehrere Computer-Netzknoten, die durch ein Netzwerk zu einer Data Warehouse- und Data Mining-Maschine miteinander verbunden sind, welche Objekte, einschließlich Bildern, Tönen und Videos, sowie einfachen und strukturierten Text indiziert. Ein Objekt wird von einer externen Datenbank über einen Knoten, der als Warehousing-Knoten bezeichnet wird, vom Netzwerk heruntergeladen. Der Warehousing-Knoten extrahiert einige Merkmale aus dem Objekt, fragmentiert die einzelnen extrahierten Merkmale in eine Anzahl an Merkmalsfragmenten, und streuspeichert diese Merkmalsfragmente. Jedes streugespeicherte Merkmalsfragment wird zu einem Knoten im Netzwerk übertragen, der als Indexknoten bezeichnet wird. Jeder Knoten im Netzwerk, der ein streugespeichertes Merkmalsfragment empfängt, verwendet das streugespeicherte Merkmalsfragment des Objekts, um eine Suche in der jeweiligen Partition der Indexdatenbank durchzuführen. Die Ergebnisse der Suchen in den lokalen Datenbanken werden vom Warehousing-Knoten gesammelt. Der Warehousing-Knoten verwendet diese Ergebnisse, um zu bestimmen, ob das Objekt bereits im Data Warehouse indiziert wurde. Danach extrahiert der Warehousing-Knoten die Merkmale aus dem Objekt, fragmentiert die Merkmale und streuspeichert diese Merkmalsfragmente. Jedes streugespeicherte Merkmalsfragment wird zu einem Knoten im Netzwerk übertragen. Jeder Knoten im Netzwerk, der ein streugespeichertes Merkmalsfragment empfängt, verwendet das

streugespeicherte Merkmalsfragment des Objekts, um das Merkmal in seiner jeweiligen Partition der Indexdatenbank zu speichern.

Bei der Abfrage kann es sich zum Beispiel um eine Musterabfrage handeln. Eine Musterabfrage ist eine Suche nach einem Muster in den Daten. Eine Musterabfrage wird von einem Anwender an einen der Front-End-Computer übertragen, der die Musterabfrage an einen der Indexknoten, welcher als Heimknoten bezeichnet wird, der Data Mining-Maschine weiterleitet. Der Heimknoten zerlegt die Musterabfrage in eine oder mehrere Unterabfragen, wobei jede Unterabfrage im Speicher gespeichert wird und ein Objektmerkmal enthält, und ein vom Computer ausführbares Programm implementiert ein Verfahren, wie zum Beispiel eine Berechnung. Die Berechnung kann zusätzlich Unterabfragen umfassen. Der Heimknoten fragmentiert die Merkmale der einzelnen Unterabfragen in ein oder mehrere Unterabfragemerkmalsfragmente und streuspeichert danach die Merkmalsfragmente. Jedes Unterabfragemerkmalsfragment wird gemäß dem streugespeicherten Merkmalsfragment an einen Knoten im Netzwerk übertragen. Jeder Knoten im Netzwerk, der eine Unterabfrage empfängt, verwendet das streugespeicherte Merkmalsfragment der Unterabfrage, um eine Suche auf der jeweiligen Partition der Indexdatenbank durchzuführen, und die Daten, auf die dabei zugegriffen wird, werden bei der Berechnung der Unterabfrage verwendet. Wenn die Berechnung einer Unterabfrage zusätzliche Unterabfragen enthält (und sie kann null, eine oder mehrere Unterabfragen enthalten), werden die zusätzlichen Unterabfragen rekursiv evaluiert, und die von der rekursiven Evaluierung erhaltenen Daten werden bei der Berechnung der Unterabfrage verwendet. Die Ergebnisse der Suchen der lokalen Indexdatenbanken und die Ergebnisse eventueller rekursiver Evaluierungen werden vom Heimknoten gesammelt. Die Ergebnisse der Musterabfrage werden vom Heimknoten bestimmt und dem Anwender zurückgegeben.

In einem anderen Aspekt der Erfindung umfaßt ein verteiltes Computer-Datenbanksystem einen oder mehrere Front-End-Computer und einen oder mehrere Computerknoten, die durch ein Netzwerk miteinander verbunden sind, um als Wissensextraktionsmaschine zu fungieren, die sowohl die Data Warehouse-Aktivität als auch die Data Mining-Aktivität unterstützt.

Betrachten wir zuerst die Data Warehousing-Aktivität. Das Herunterladen von Objekten von einer anderen Datenbank zum Warehouse wird von einem Warehouse-Knoten durchgeführt. Hinsichtlich eines Objekts, das von einer anderen Datenbank heruntergeladen wird, bestimmt der Warehousing-Knoten zuerst, ob das Objekt aufgrund eines Downloads von einer anderen Datenbank möglicherweise bereits im Data Warehouse repräsentiert wird. Wenn dies der Fall ist, extrahiert der Warehouse-Knoten ein oder mehrere Merkmale des Objekts, fragmentiert die einzelnen Objektmerkmale in eine Anzahl an Merkmalsfragmenten und streuspeichert danach diese einzelnen Merkmalsfragmente. Ein Anteil eines jeden streugespeicherten Fragments wird vom Warehouse-Knoten als Adressierindex verwendet, durch den der Warehouse-Knoten das streugespeicherte Abfragemerkmal an einen Indexknoten des Netzwerks überträgt. Jeder Indexknoten im Netzwerk, der ein streugespeichertes Objektmerkmalsfragment empfängt, verwendet das streugespeicherte Objektmerkmalsfragment, um eine Suche in der jeweiligen Indexdatenbank durchzuführen. Knoten, die Daten finden, welche dem streugespeicherten Objektmerkmal entsprechen, geben die OIDs der Warehouse-Objekte, welche dieses Merkmalsfragment enthalten, zurück. Solche OIDs werden dann vom Warehouse-Knoten gesammelt, und es wird eine Ähnlichkeitsfunktion berechnet. Diese Ähnlichkeitsfunktion wird verwendet, um zu bestimmen, ob das Objekt bereits im Data Warehouse gespeichert ist. Wenn festgestellt wird, daß das Objekt im Data Warehouse repräsentiert wird, wird die OID des Warehouse-Objekts für das heruntergeladene Objekt verwendet. Wenn es noch nicht repräsentiert wird, wird eine einzigartige OID für das Objekt ausgewählt. Danach extrahiert der Warehousing-Knoten Merkmale aus dem Objekt, fragmentiert diese und streuspeichert diese Merkmalsfragmente. Ein Anteil eines jeden streugespeicherten Fragments wird vom Warehouse-Knoten als Adressierindex verwendet, durch den der Warehouse-Knoten das streugespeicherte Objektabfragemerkmal an einen Indexknoten des Netzwerks überträgt, wo das Merkmal im Data Warehouse gespeichert wird.

Betrachten wir als nächstes die Data Mining-Aktivität. Ein Anwender, der eine Abfrage evaluieren möchte, wie zum Beispiel eine Suche nach einem Muster in den Daten durchführen möchte, überträgt eine Abfrage zu einem der Front-End-Computer, der wiederum die Abfrage an einen der Indexknoten im Netzwerk weiterleitet. Der Knoten, der die Abfrage empfängt (er wird als Heimknoten des Data Warehouse bezeichnet), zerlegt die Abfrage in eine oder mehrere Unterabfragen. Eine Unterabfrage umfaßt ein Merkmal und ein vom Computer ausführbares Programm, das ein Verfahren implementiert, wie zum Beispiel eine Berechnung, welche zusätzliche Unterabfragen umfassen kann. Der Heimknoten speichert diese und fragmentiert die Merkmale einer jeden Unterabfrage in ein oder mehrere Unterabfragemerkmalsfragmente, und streuspeichert danach die einzelnen Merkmalsfragmente der Unterabfragen. Ein Anteil eines jeden streugespeicherten Merkmalsfragments wird vom Heimknoten als Adressierindex verwendet, durch den der Heimknoten die streugespeicherte Abfrage an einen Knoten des Netzwerks überträgt. Jeder Indexknoten im Netzwerk, der eine Unterabfrage empfängt, verwendet das streugespeicherte Merkmal, um eine Suche in der jeweiligen Indexdatenbank durchzuführen. Knoten, die Daten finden, welche dem streugespeicherten Merkmalsfragment der Unterabfrage entsprechen, führen die in der Unterabfrage festgelegte Berechnung durch. Wenn die Berechnung keine zusätzlichen Unterabfragen enthält, werden die Ergebnisse der Berechnung an den Heimknoten zurückgegeben. Wenn die Berechnung jedoch zusätzliche Unterabfragen enthält, übernimmt der Knoten die Rolle des Heimknotens im Hinblick auf die in der Berechnung enthaltenen Unterabfragen. Insbesondere streuspeichert der Konten die Merkmalsfragmente der enthaltenen Unterabfragen und überträgt die Unterabfragen zu anderen Knoten. Dieser Prozeß wird rekursiv fortgesetzt, bis die Berechnung vollständig ist, und die endgültigen Ergebnisse werden an den ursprünglichen Heimknoten zurückgesandt. Bei Empfang der Ergebnisse der Berechnung führt der Heimknoten eventuell noch verbleibende Datenaggregationen durch, die von der ursprünglichen Musterabfrage festgelegt wurden, und überträgt die Informationen zum Front-End-Knoten. Der Front-End-Knoten formatiert die Antwort an den Anwender und überträgt die formatierte Antwort zum Anwender.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

Die oben genannten sowie weitere Vorteile der Erfindung können besser durch die Bezugnahme auf die folgende Beschreibung in Verbindung mit den begleitenden Zeichnungen verstanden werden, in denen:

Fig. 1 ein Blockdiagramm einer Ausführungsform des verteilten Computer-Datenbanksystems gemäß der Erfindung ist;

Fig. 2 ein Blockdiagramm des verteilten Computer-Da-

tenbanksystems von **Fig.** 1 in Ablaufdiagramm-Form ist, welches ein Verfahren zum Herunterladen von Informationen von einer anderen Quelle zum Data Warehouse gemäß einer Ausführungsform der Erfindung darstellt;

Fig. 3 ein Blockdiagramm des verteilten Computer-Datenbanksystems von **Fig.** 1 in Ablaufdiagramm-Form ist, welches ein Verfahren zur Beantwortung einer Abfrage gemäß einer Ausführungsform der Erfindung darstellt;

Fig. 4a–**Fig.** 4e Blockdiagramme sind, welche Formate für eine Warehouse-Meldung, eine Warehouse-Antwort-Meldung, eine Einfüge-Meldung, eine Unterabfrage-Meldung bzw. eine Unterabfrage-Antwort-Meldung zeigen, wie sie in Verbindung mit der Ausführungsform von **Fig.** 1–3 verwendet werden können;

Fig. 5 ein Blockdiagramm eines der Heimknoten von **Fig.** 1–3 gemäß einer Ausführungsform der Erfindung ist;

Fig. 6 ein Blockdiagramm eines Indexknotens von **Fig.** 1–3 gemäß einer Ausführungsform der Erfindung ist; und

Fig. 7 ein Blockdiagramm eines Computersystem gemäß einer beispielhaften Ausführungsform eines Anwendercomputers, eines Indexknotens bzw. eines Warehouse-Knotens ist.

DETAILLIERTE BESCHREIBUNG DER BEVORZUGTEN AUSFÜHRUNGSFORM

Bezugnehmend auf **Fig.** 1 umfaßt eine Ausführungsform eines verteilten Computer-Datenbanksystems **100** gemäß der Erfindung einen Anwendercomputer **102**, der z. B. über ein Netzwerk **106** mit einem Front-End-Computer **104** in Verbindung steht. Alternativ dazu kann es sich bei dem Front-End-Computer **104** auch um den Anwender-Computer handeln. Der Front-End-Computer **104** steht wiederum in Verbindung mit einer Data Warehouse- und Data Mining-Maschine, die einen oder mehrere Computerknoten **106**, **108** umfaßt, die durch ein lokales Netzwerk **110** miteinander verbunden sind. Die einzelnen Computerknoten **106**, **108** können lokale Festplatten **112** umfassen, oder sie können alternativ oder zusätzlich dazu Daten von einem Netzwerk-Festplattenserver (nicht abgebildet) erhalten.

Bei den Computerknoten **106**, **108** des Data Warehouse kann es sich um verschiedene Arten, wie zum Beispiel Indexknoten **106** und Warehouse-Knoten **108**, handeln. Die Knoten **106**, **108** des Data Warehouse müssen nicht unterschiedliche Computer repräsentieren. In einer Ausführungsform handelt es sich bei dem Data Warehouse um einen einzelnen Computer, der die Rolle aller Indexknoten **106** und Warehouse-Knoten **108** übernimmt. In einer anderen Ausführungsform wird das Data Warehouse durch separate Computer für jeden Indexknoten **106** und jeden Warehouse-Knoten **108** dargestellt.

Fachleute dieses Bereiches werden anerkennen, daß zahlreiche Variationen möglich sind, die jedoch alle innerhalb des Umfangs und Geistes der vorliegenden Erfindung liegen.

Wenn wir ein beispielhaftes Verfahren **200** betrachten, bei dem die Objekte zuerst heruntergeladen werden, und indem wir auch auf **Fig.** 2 Bezug nehmen, sehen wir, daß in einer Ausführungsform Objekte von einer externen Datenbank **201** durch einen oder mehrere Warehouse-Knoten **108** heruntergeladen werden (Schritt **201**). Wenn ein Objekt aufgrund eines früheren Downloads, z. B. von einer anderen Datenbank, bereits im Data Warehouse repräsentiert wird, extrahiert der Warehouse-Knoten **108** eine Anzahl an Merkmalen aus dem Objekt, um das Objekt zu identifizieren, wie dies im integrierten Datenmodell des Data Warehouses festgelegt ist. Zum Beispiel kann eine Person durch eine Mitarbeiter-"ID", eine Kontonummer, Name, Adresse, Telefonnummer, E-Mail-Adresse, usw. oder durch eine beliebige Kombination aus diesen identifiziert werden.

Es kann eine Vielzahl unterschiedlicher Extraktionstechniken verwendet werden. Für relationale Attributwerte, wie zum Beispiel das Datum einer Transaktion, können die möglichen Werte in eine Sammlung von aneinander angrenzenden, nicht überlappenden Bereichen aufgeteilt werden. Das derartige Aufteilen von Feldwerten wird als Diskretisieren bezeichnet. Der tatsächliche Wert kann auch im Indexeintrag enthalten sein.

Merkmale werden aus strukturierten Dokumenten extrahiert, indem das Dokument "geparst" wird (das heißt, es wird eine automatische Syntaxanalyse durchgeführt), um eine Datenstruktur zu erstellen. Danach wird diese Datenstruktur in (möglicherweise überlappende) Substrukturen unterteilt, die als Fragmente bezeichnet werden. Das einer Unterabfrage zugeordnete Fragment wird dazu verwendet, um übereinstimmende Fragmente in der Datenbank zu finden; es wird daher als Muster bezeichnet.

Merkmale, die aus unstrukturierten Dokumenten extrahiert werden, werden in einer Datenstruktur strukturiert, die eine Sammlung von untereinander in Beziehung stehenden Substrukturen umfaßt, welche danach in (möglicherweise überlappende) Komponenten-Substrukturen unterteilt werden, wie im Falle eines strukturierten Dokuments, und diese Komponenten-Substrukturen sind die Fragmente des unstrukturierten Dokuments.

Für Medien wie z. B. Töne, Bilder und Videos wurde eine große Vielzahl unterschiedlicher Merkmalsextraktionsalgorithmen entwickelt, wie zum Beispiel Kantenerkennungs-, Segmentierungs- und Objektklassifizierungsalgorithmen für Bilder. Fourier- und Wavelet-Transformationen sowie zahlreiche Filteralgorithmen werden ebenfalls verwendet, um Merkmale aus Bildern und Tönen zu extrahieren. Merkmale können auch manuell oder halbautomatisch zu einem Objekt hinzugefügt werden. Solche hinzugefügten Merkmale werden als Annotationen oder Metadaten bezeichnet. Merkmale werden aus Annotationen mit Hilfe einer der oben erwähnten Techniken extrahiert. Dies hängt davon ab, ob es sich bei der Annotation um einen Eintrag einer relationalen Datenbank, ein strukturiertes Dokument oder ein unstrukturiertes Dokument handelt. Wenn einem Merkmal Werte zugeordnet sind, können sie diskretisiert werden. Es ist auch möglich, Beziehungen zwischen Merkmalen festzulegen. So kann zum Beispiel ein Merkmal innerhalb eines anderen Merkmals enthalten sein oder sich neben einem anderen Merkmal befinden. Das integrierte Datenmodell spezifiziert die Merkmalsextraktionsalgorithmen sowie die Struktur der Merkmale.

Der Warehouse-Knoten **108** codiert jedes Merkmalsfragment des Objekts durch Verwendung einer vordefinierten Streuspeicherfunktion. Daten im System wurden zuvor mit Hilfe dieser Streuspeicherfunktion lokal auf den verschiedenen Indexknoten gespeichert, um einen Index zu den Daten in der lokalen Datenbank zu erzeugen. Somit stellt die Verwendung der selben Streuspeicherfunktion zur Erzeugung eines Indexes für die Datenspeicherung und zur Erzeugung von streugespeicherten Mustern für ein Objekt sicher, daß Daten während des Speicherns von Daten gleichmäßig über die Indexknoten **106** des Data Warehouse verteilt werden.

In einer Ausführungsform besitzt der sich aus der Verwendung der Streuspeicherfunktion ergebende Streuspeicherwert einen ersten Abschnitt, der dazu dient, den Indexknoten zu identifizieren, an den Daten für die Speicherung gesendet werden sollen oder an den ein Merkmalsfragment als Muster zu senden ist. Der Streuspeicherwert besitzt auch einen zweiten Abschnitt, der als lokaler Indexwert bezeichnet wird, und der dazu verwendet wird, um die Speicherpo-

sitionen zu bestimmen, an denen Daten zu speichern sind oder von denen Daten vom Indexknoten zu holen sind. Somit werden die streugespeicherten Objektmerkmalsfragmente (Schritt **202**) als Muster an bestimmte Indexknoten **106** des Data Warehouse verteilt, welche durch den ersten Abschnitt des streugespeicherten Wertes bestimmt werden.

Die Indexknoten **106**, deren Muster mit den streugespeicherten Merkmalsfragmenten übereinstimmen, durch welche die Daten ursprünglich am Indexknoten gespeichert wurden, antworten auf eine Wiedergewinnungsmeldung durch die Übertragung (Schritt **203**) der OIDs, welche den streugespeicherten Merkmalsfragmenten der angeforderten Informationen entsprechen, zum Warehouse-Knoten **108**. Somit werden sämtliche Übereinstimmungen zwischen den streugespeicherten Mustern und einer lokalen Streuspeichertabelle der streugespeicherten Merkmalsfragmente zurückgegeben oder am Warehouse-Knoten **108** gesammelt, der die Objektmerkmalsfragmente anfänglich streugespeichert hat.

Der Warehouse-Knoten **108** bestimmt danach, ob eine der OIDs das selbe Objekt repräsentiert wie das im Warehouse zu verarbeitende Objekt. Diese Bestimmung wird vom Warehouse-Knoten durch Vergleich des Ähnlichkeitsgrades zwischen dem im Warehouse zu verarbeitenden Objekt und den Objekten, deren OIDs zurückgegeben wurden, durchgeführt. In einer Ausführungsform wird das Maß der Ähnlichkeit durch die Merkmale bestimmt, die den Objekten gemein sind, und den Merkmalen des im Warehouse zu verarbeitenden Objekts, die keine Merkmale des Objekts sind, dessen OID zurückgegeben wurde.

Dieses Ähnlichkeitsmaß kann auf dem Merkmalskontrastmodell von Tversky (Referenz oben) basieren. Der erste Term trägt eine positive Zahl zum Ähnlichkeitswert bei, während der zweite einen negativen Beitrag leistet. Darüber hinaus wird der zweite Term mit einer vordefinierten Konstanten multipliziert, so daß ein Merkmal in der zweiten Gruppe weniger Auswirkungen auf die Ähnlichkeit hat als eines in der ersten Gruppe.

Wenn bestimmt wird, daß das Objekt im Data Warehouse repräsentiert wird, dann steht bereits eine OID für das Objekt bereit. Wenn es noch nicht repräsentiert wird, wird eine einzigartige OID für das Objekt ausgewählt.

Danach extrahiert der Warehouse-Knoten **108** alle Merkmale des Objekts gemäß dem integrierten Datenmodell des Data Warehouse. Die Merkmalsextraktionstechniken wurden oben diskutiert. Der Warehouse-Knoten **108** fragmentiert die einzelnen Merkmale in Merkmalsfragmente und codierte die einzelnen Merkmalsfragmente des Objekts durch Verwendung einer vordefinierten Streuspeicherfunktion, wie dies oben diskutiert wurde. In einer Ausführungsform besitzt der Streuspeicherwert, der sich aus der Verwendung der Streuspeicherfunktion ergibt, einen ersten Abschnitt, der dazu dient, den Indexknoten zu identifizieren, zu dem die zu speichernden Daten gesendet werden sollen (Schritt **204**), und einen zweiten Abschnitt, bei dem es sich um einen lokalen Indexwert handelt, der verwendet wird, um zu bestimmen, wo die Daten am Indexknoten zu speichern sind (Schritt **205**).

Betrachten wir als nächstes ein beispielhaftes Verfahren **300** für die Verarbeitung einer Abfrage und nehmen wir dazu Bezug auf **Fig.** 3. Wenn in einer Ausführungsform ein Anwender (Schritt **301**) eine Abfrage vom Anwendercomputer **102** überträgt, empfängt der Front-End-Computer **104** die Abfrage. Der Front-End-Computer **104** ist dafür verantwortlich, die Verbindung mit dem Anwendercomputer **102** aufzunehmen, um es dem Anwender zu ermöglichen, eine Abfrage zu übertragen und eine Antwort in einem entsprechenden Format zu empfangen. Der Front-End-Computer **104** ist auch für sämtliche Authentifizierungs- und Verwaltungsfunktionen verantwortlich. In einer Ausführungsform handelt es sich bei dem Front-End-Computer **104** um einen World Wide Web-Server, der mit dem Anwendercomputer **102** über das HTTP-Protokoll kommuniziert.

Nach der Überprüfung, ob die Abfrage akzeptabel ist, führt der Front-End-Computer **104** alle Neuformatierungsarbeiten durch, die notwendig sind, um die Abfrage mit den Anforderungen des Data Warehouse kompatibel zu machen. Der Front-End-Computer **104** überträgt danach die Abfrage zu einem der Indexknoten **106** des Data Warehouse (Schritt **302**), der danach als Heimknoten **107** des Data Warehouse für diese Abfrage bezeichnet wird.

Der Heimknoten **107** zerlegt die Abfrage in eine Anzahl (eine oder mehrere) von Unterabfragen. Jede Unterabfrage besitzt ein Merkmal und spezifiziert ein vom Computer ausführbares Verfahren, z. B. eine Berechnung. Die Berechnung bestimmt, welche Maßnahme die Unterabfrage auszuführen hat. Die häufigsten Berechnungen sind statistische Funktionen, die Informationen sammeln, welche im Data Warehouse gespeichert sind. Berechnungen können Ähnlichkeitskriterien wie zum Beispiel die zur Akzeptierung einer Übereinstimmung erforderliche Mindeststärke und statistische Berechnungen, wie zum Beispiel den Durchschnitt oder die Standardabweichung, umfassen. Die Berechnung kann zusätzliche Unterabfragen umfassen.

Für jede Unterabfrage fragmentiert der Heimknoten **107** das Unterabfragemerkmal in Unterabfragemerkmalsfragmente und codiert das Merkmalsfragment durch Verwendung einer vordefinierten Streuspeicherfunktion, wie dies oben beschrieben ist. Das streugespeicherte Fragment und die Unterabfrage werden unter Verwendung des streugespeicherten Merkmalsfragments wie oben beschrieben vom Heimknoten zu einem Indexknoten übertragen (Schritt **303**).

Der Indexknoten **106**, dessen streugespeicherte Fragmente mit den Indexmerkmalsfragmenten übereinstimmen, durch welche die Daten anfänglich an jenem Indexknoten gespeichert wurden, reagieren auf die Unterabfragen, indem sie Daten in die lokale Streuspeichertabelle von Indextermini holen, welche mit dem streugespeicherten Merkmalsfragment übereinstimmen, und indem sie die in der Unterabfrage angegebene Berechnung durchführen. Wenn die Berechnung zusätzliche Unterabfragen enthält, übernimmt der Indexknoten die Funktion eines Heimknotens für eine neue Abfrage, die als Komponentenunterabfrage bezeichnet wird, welche wie oben beschrieben verarbeitet wird (Schritt **304**). Zum Beispiel könnte eine Unterabfrage verwendet werden, um andere Produktumsätze zu finden, die mit einzelnen Kunden im Zusammenhang stehen, welche im letzten Monat ein Gerät gekauft haben. Unabhängig davon, ob die Berechnung zusätzliche Unterabfragen enthält oder nicht, gibt der Indexknoten die Ergebnisse seiner Berechnung an den Heimknoten **107** der Unterabfrage zurück, der sie erhalten hat (Schritt **305**).

Wenn die Ergebnisse aller Unterabfragen der ursprünglichen Abfrage empfangen wurden, führt der Heimknoten **107** sämtliche Datenaggregationen durch, wie zum Beispiel die Berechnung des Durchschnitts oder der Standardabweichung, die von der ursprünglichen Abfrage angegeben wurden, und gibt die sich daraus ergebenden Informationen an den Anwender zurück. In einer Ausführungsform werden die zurückgegebenen Informationen an den Front-End-Computer **104** übertragen (Schritt **306**), der die Antwort entsprechend formatiert und die Antwort an den Anwender überträgt (Schritt **307**). In einer anderen Ausführungsform werden die zurückzugebenden Informationen ohne Intervention des Front-End-Computers **104**, z. B. über ein Netzwerk **105**, direkt zum Anwendercomputer **102** übertragen.

Als nächstes werden die in der bevorzugten Ausführungsform verwendeten Meldungsformate besprochen und dabei auf **Fig.** 4a Bezug genommen. Ein beispielhaftes Format für eine Warehouse-Meldung umfaßt vier Felder: die Kopfzeile **402**, den Objektbezeichner (QID) **403**, das streugespeicherte Objektfragment (HOF) **404**, und den Wert **405**. Das Kopfzeilenfeld **402** gibt an, daß es sich bei dieser Meldung um eine Warehouse-Meldung handelt, und es gibt auch den Bestimmungsindexknoten an. Der Bestimmungsindexknoten wird vom ersten Abschnitt des streugespeicherten Objektfragments bestimmt. Das OID-Feld **403** enthält einen Objektart-Spezifizierer und einen Objektbezeichner. Das HOF-Feld **404** enthält einen Fragmentart-Spezifizierer und den zweiten Abschnitt des streugespeicherten Objektfragments, das vom Streuspeichermodul erzeugt wird (**Fig.** 5). Das Wert-Feld **405** enthält einen wahlweisen Wert, der dem Fragment zugeordnet ist. Der Fragmentart-Spezifizierer bestimmt, ob die Warehouse-Meldung ein Wert-Feld **405** enthält, und wenn die Warehouse-Meldung tatsächlich ein Wert-Feld enthält, bestimmt der Fragmentart-Spezifizierer die Größe des Wert-Feldes.

Bezugnehmend auf **Fig.** 4b besitzt ein beispielhaftes Format einer Warehouse-Antwortmeldung zwei Teile: den Bezeichner und Werte. Der Bezeichner-Teil besitzt vier Felder: Kopfzeile **406**, OID1 **407**, OID2 **408**, und Gewicht **409**. Das Kopfzeilenfeld **406** gibt an, daß es sich bei dieser Meldung um eine Warehouse-Antwortmeldung handelt, und es gibt auch den Warehouse-Bestimmungsknoten an. Der Warehouse-Bestimmungsknoten ist der Warehouse-Knoten, von dem die entsprechende Warehouse-Meldung empfangen wurde. Die beiden OID-Felder **407**, **408** enthalten einen Objektart-Spezifizierer und einen Objektbezeichner. Das erste OID-Feld **407** ist gleich wie das OID-Feld **403** der entsprechenden Warehouse-Meldung. Das zweite OID-Feld **408** identifiziert ein Objekt, das zuvor indiziert wurde. Das Gewicht-Feld **409** enthält ein optionales Gewicht, das dem Objekt zugeordnet ist, welches durch OID1 **407** identifiziert wird. Der Objektart-Spezifizierer von OID1 bestimmt, ob die Warehouse-Antwortmeldung ein Gewicht-Feld enthält, und wenn die Warehouse-Abfrageantwortmeldung tatsächlich ein Gewicht-Feld enthält, bestimmt der Objektart-Spezifizierer von OID1 die Größe des Feldes. Der Werte-Teil der Warehouse-Antwortmeldung enthält eine Anzahl von Feldern **410**, welche Daten enthalten, die dem von OID2 **408** identifizierten Objekt zugeordnet sind. Die Struktur und Größe des Werte-Teiles wird vom Objekttyp-Spezifizierer von OID2 bestimmt.

Bezugnehmend auf **Fig.** 4c besitzt ein beispielhaftes Format für eine Einfügemeldung vier Felder: Kopfzeile **411**, OID **412**, HOF **413**, und Wert **414**. Das Kopfzeilen-Feld **414** gibt an, daß es sich bei dieser Meldung um eine Einfügemeldung handelt, und es legt auch den Bestimmungsindexknoten fest. Der Bestimmungsindexknoten wird vom ersten Abschnitt des streugespeicherten Objektfragments bestimmt. Das OID-Feld **412** enthält einen Objektart-Spezifizierer und einen Objektbezeichner. Das HOF-Feld **413** enthält einen Fragmentart-Spezifizierer und den zweiten Abschnitt des streugespeicherten Objektfragments, das vom Streuspeichermodul erzeugt wird (**Fig.** 5). Das Wert-Feld **414** enthält einen wahlweisen Wert, der dem Fragment zugeordnet ist. Der Fragmentart-Spezifizierer bestimmt, ob die Einfügemeldung ein Wert-Feld **414** enthält, und wenn die Einfügemeldung tatsächlich ein Wert-Feld enthält, bestimmt der Fragmentart-Spezifizierer die Größe des Wert-Feldes.

Bezugnehmend auf **Fig.** 4d besitzt ein beispielhaftes Format einer Unterabfragemeldung zwei Teile: den Bezeichner und Unterabfragen. Der Bezeichner-Teil besitzt vier Felder: die Kopfzeile **415**, den Unterabfragebezeichner (QSID) **416**, das streugespeicherte Abfragefragment (HQF) **417**, und den Wert **418**. Das Kopfzeilenfeld **415** gibt an, daß es sich bei dieser Meldung um eine Unterabfragemeldung handelt, und es gibt auch den Bestimmungsindexknoten an. Der Bestimmungsindexknoten wird vom ersten Abschnitt des streugespeicherten Abfragefragments bestimmt. Das QSID-Feld **416** enthält einen Abfrageart-Spezifizierer und einen Unterabfragebezeichner. Das HQF-Feld **417** enthält einen Fragmentart-Spezifizierer und den zweiten Abschnitt des streugespeicherten Unterabfragefragments, das vom Streuspeichermodul erzeugt wird (**Fig.** 5). Das Wert-Feld **418** enthält einen wahlweisen Wert, der dem Fragment zugeordnet ist. Der Fragmentart-Spezifizierer bestimmt, ob die Unterabfragemeldung ein Wert-Feld **418** enthält, und wenn die Unterabfrage-Meldung tatsächlich ein Wert-Feld enthält, bestimmt der Fragmentart-Spezifizierer die Größe des Wert-Feldes. Der Unterabfragen-Teil der Unterabfragemeldung enthält eine Anzahl an Unterabfragen. Eine Unterabfragemeldung, welche keine Unterabfragen besitzt, wird als Einfache Unterabfragemeldung bezeichnet.

Bezugnehmend auf **Fig.** 4e besitzt eine beispielhafte Ausführungsform einer Unterabfrageantwortmeldung zwei Teile: den Bezeichner und Werte. Der Bezeichner-Teil besitzt zwei Felder: die Kopfzeile **420** und die QSID **421**. Das Kopfzeilenfeld **420** gibt an, daß es sich bei dieser Meldung um eine Unterabfrageantwortmeldung handelt, und es gibt auch den Bestimmungsindexknoten an. Der Bestimmungsindexknoten ist der selbe wie der Indexknoten, von dem die entsprechende Unterabfragemeldung empfangen wurde. Das QSID-Feld **421** enthält einen Abfrageart-Spezifizierer und einen Unterabfragebezeichner. Der Werte-Teil der Unterabfrageantwortmeldung besitzt eine Anzahl an Feldern **422**, welche die Ergebnisdaten der Unterabfrage aufnehmen. Die Struktur des Werte-Teiles wird vom Abfrageart-Spezifizierer spezifiziert.

Jeder Knoten des verteilten Computersystems umfaßt ein Kommunikationsmodul, das im folgenden diskutiert wird und in **Fig.** 5 und 6 dargestellt ist, und das für das Übertragen und Empfangen von Meldungen zwischen zwei Knoten verantwortlich ist. Die Übertragung einer Meldung erfordert (1) das In-die-Warteschlange-Stellen der Meldung vor der Übertragung über das Kommunikationsmedium, (2) die tatsächliche Übertragung über das Kommunikationsmedium, und (3) das In-die-Warteschlange-Stellen einer Aufgabe, um die Meldung zu verarbeiten, wenn die Meldung vom Modul empfangen wird, das von der Meldungsart bestimmt wird. Die Meldungsart bestimmt den Befehl, der an das empfangende Modul geschickt wird. Der Befehl bestimmt das Mittel, durch welches die Meldung vom Modul verarbeitet werden soll. Der Bestimmungsknoten für eine zu übertragende Meldung wird im Kopfzeilen-Feld einer jeden Meldung angegeben. Wenn eine Meldung von einem anderen Knoten empfangen wird, bestimmt die Art der Meldung, welches Modul die Meldung verarbeiten wird. Die Meldungsart wird im Kopfzeilen-Feld einer jeden Meldung angegeben. Das Kommunikationsmodul eines Heimknotens ist auch für die Kommunikation mit den Front-End-Knoten verantwortlich. Ein Front-End-Knoten überträgt Abfragen zum Heimknoten, und der Heimknoten überträgt die Ergebnisse, wie zum Beispiel Graphen und formatierte Tabellen, zum Front-End-Knoten.

Als nächstes betrachten wir beispielhafte Ausführungsformen der oben diskutierten Knoten, wobei zu diesem Zweck auch auf **Fig.** 5 Bezug genommen wird. Ein Warehouse-Knoten **500** kann einen Downloader **502** besitzen, der externe Datenbanken abtastet, um Objekte für die Warehouse-Verarbeitung und Indizierung durch die Wissensextraktionsmaschine herunterzuladen. Jeder Warehouse-Kno-

ten **500** kann eine unterschiedliche Art eines Downloaders **500** besitzen. Zum Beispiel kann eine Art eines Downloaders Daten von relationalen Datenbanken mit Hilfe eines standardmäßigen SQL-Protokolls, wie zum Beispiel eines ODBC- oder eines proprietären Protokolls, das von einem Anbieter relationaler Datenbanken festgelegt wurde, herunterladen. Das Herunterladen wird in diesem Fall mit Hilfe einer oder mehrerer SQL-Abfragen durchgeführt. Für ein anderes Beispiel kann es sich bei dem Downloader um einen Informations- und Inhaltsaustauschabonnenten (ICE) handeln, der verhandelt, um Inhalte von Syndikatoren über das Internet zu erhalten. Dies ist ein bevorzugter Mechanismus zum Erlangen von zeitkritischem Inhalt, wie zum Beispiel Nachrichten. Der Downloader **502** überträgt Objekte zu einem Merkmalsextraktor **504**.

Der Merkmalsextraktor **504** extrahiert Merkmale von einem Objekt. Wenn es sich bei dem Objekt um einen Eintrag in einer relationalen Datenbank handelt, umfaßt die Merkmalsextraktion solche Schritte wie das Auswählen der Felder, die zu indizieren sind, das Neuformatieren der Felder und das Eliminieren oder Korrigieren von Daten, die als fehlerhaft bestimmt werden. Die Merkmalsextraktion für Bilder wird durch Erkennung der Kanten, Identifizierung der Bildobjekte und Bestimmung der Beziehungen zwischen Bildobjekten durchgeführt. In einer anderen Ausführungsform wird die Merkmalsextraktion für Bilder durch Berechnung der Fourier- und Wavelet-Transformationen durchgeführt. Jede Fourier- oder Wavelet-Transformation stellt ein extrahiertes Merkmal dar. Merkmale werden mit Hilfe einer Anzahl an Einfügemeldungen indiziert.

Der Merkmalsextraktor **504** bildet auch jeden Objektbezeichner in einer externen Datenbank an einen Objektidentifizierer der Wissensextraktionsmaschine ab. Jede externe Datenbank kann ihren eigenen Mechanismus zur Zuweisung von Objektidentifizierern besitzen, und Merkmale des selben Objekts können in jeder externen Datenbank mit einem unterschiedlichen Objektidentifizierer gespeichert werden. Zum Beispiel kann eine externe Datenbank eine Sozialversicherungsnummer verwenden. Eine andere Datenbank könnte eine Mitarbeiterkennung verwenden. Die Abbildung von externen Objektidentifizierern wird durch Verwendung einer Anzahl von Warehouse-Meldungen erzielt.

Ein Fragmentierer **506** berechnet die in den einzelnen Merkmalen enthaltenen Fragmente. Jedes Fragment besteht aus einer abgegrenzten Gruppe miteinander in Beziehung stehender Komponenten des Merkmals. In einer Ausführungsform umfassen die Fragmente eines Merkmals jedes Attribut und jede Beziehung in der Datenstruktur, welche das Merkmal definieren. Bei einem Objekt in der Form eines Eintrags einer relationalen Datenbank handelt es sich bei den Merkmalen um die Attribute, die vom Merkmalsextraktor **504** ausgewählt, neu formatiert und korrigiert wurden. Die Fragmente werden zum Streuspeichermodul übertragen.

Ein Streuspeichermodul **508** berechnet eine Streuspeicherfunktion eines Fragments. In einer Ausführungsform handelt es sich bei der Streuspeicherfunktion um den MD4 Message Digest Algorithmus, der in einer Spezifikation, Request for Comment (RFC) 1185, veröffentlicht von der Network Working Group der Internet Engineering Task Force, Oktober 1990, beschrieben ist, und über das Internet oder von R. Rivest am MIT Laboratory for Computer Science, Cambridge, MA, USA, erhältlich ist. Das Streuspeichermodul **508** überträgt entweder eine Warehouse-Meldung oder eine Einfügemeldung zu einem Kommunikationsmodul **510**; dies hängt davon ab, ob der Zweck des Fragments darin besteht, eine Objektidentifizierer-Abbildung zu erzielen, oder ein Objektmerkmal zu indizieren.

Ein Ähnlichkeitskomparator **512** empfängt Warehouse-Antwortmeldungen vom Kommunikationsmodul **510** und erzeugt Einfügemeldungen, die zum Kommunikationsmodul **510** übertragen werden. Der Ähnlichkeitskomparator **512** sammelt alle Warehouse-Antworten für ein Objekt, dessen Bezeichner abgebildet wird. Für jedes Objekt in den Antworten bestimmt der Ähnlichkeitskomparator **512** die Relevanz eines jeden in der Suche zurückgegebenen Objektbezeichners. Diese Bestimmung der Relevanz wird vom Warehouse-Knoten durch Vergleich des Ähnlichkeitsgrades zwischen dem Objekt, dessen Bezeichner abgebildet wird, und den Objekten, deren OIDs zurückgegeben wurden, durchgeführt. In einer Ausführungsform ist das Maß der Ähnlichkeit zwischen der Abfrage und dem Objekt ein Cosinusmaß, das vom Ausdruck COS(v, w) angegeben wird, wobei der Vektor v die Abfrage bezeichnet, und der Vektor w das Objekt bezeichnet. Diese Vektoren befinden sich in einem Raum, in dem jedes Fragment eine Dimension des Raumes repräsentiert. Wenn eine kompatible OID gefunden wird, wird die OID als abgebildeter Objektbezeichner verwendet, und die OID wird zum Merkmalsextraktor **504** übertragen. Wenn keine kompatible OID gefunden wird, wird ein neuer Objektbezeichner ausgewählt und zum Merkmalsextraktor **504** übertragen.

Bezugnehmend auf **Fig.** 6 kann ein Indexknoten **600** ein Fragmenttabellenmodul **602** besitzen, das Warehouse-Meldungen, Einfügemeldungen und Einfache Unterabfragemeldungen von einem Kommunikationsmodul **604** empfängt. Im Falle einer Warehouse-Meldung holt das Fragmenttabellenmodul **602** einen Eintrag in eine lokale Streuspeichertabelle **603**, wofür der Streuspeicherwert im HOF-Feld verwendet wird. Der Arten-Spezifizierer im HOF-Feld und der Eintrag in der lokalen Streuspeichertabelle werden zu einem Fragmentkomparator **606** übertragen. Im Falle einer Einfachen Unterabfrage-Meldung holt das Fragmenttabellenmodul **602** einen Eintrag in eine lokale Streuspeichertabelle **603**, wofür der Streuspeicherwert im HQF-Feld verwendet wird. Der Eintrag in der lokalen Streuspeichertabelle **603** wird mit Hilfe einer Unterabfrageantwortmeldung an einen Abfrageprozessor **608** zurückgegeben. Im Falle einer Einfügemeldung modifiziert das Fragmenttabellenmodul **602** einen Eintrag in der lokalen Streuspeichertabelle **603** durch Einfügung der OID- und Wert-Felder der Einfügemeldung in den Eintrag in der lokalen Streuspeichertabelle **603**.

Der Fragmentkomparator **606** empfängt Einträge aus dem Fragmenttabellenmodul **602**. Eine Vergleichsfunktion wird vom HOF-Arten-Spezifizierer bestimmt, der vom Fragmenttabellenmodul **602** übertragen wurde. Die Vergleichsfunktion wird dazu verwendet, um die Relevanz der OID- und Wert-Felder im Eintrag zu bestimmen, der vom Fragmenttabellenmodul **602** übertragen wurde. In einer Ausführungsform bestimmt die Vergleichsfunktion ein Ähnlichkeitsgewicht, und die OIDs mit dem höchsten Ähnlichkeitsgewicht werden als relevant erachtet. Die relevanten OIDs und deren Ähnlichkeitsgewichte werden mit Hilfe einer Warehouse-Antwortmeldung zum Kommunikationsmodul **604** übertragen.

Ein Abfrage-Parser **612** führt ein Parsing an einer Abfrage in einem Abfrageberechnungsbaum, der im Speicher **613** gespeichert ist, durch, wobei es sich um eine Datenstruktur handelt, die hinsichtlich einer Anzahl an Knoten und ihrer Beziehungen zueinander spezifiziert ist. Die Knoten der Abfrageberechnungsbäume sind entweder interne Knoten oder Blattknoten. Ein interner Knoten ist ein Knoten mit einem oder mehreren Töchterknoten. Ein interner Knoten legt fest, wie die Ergebnisse der Töchterknoten zu kombinieren sind. Zum Beispiel könnten die Summen summiert oder gemittelt oder zur Berechnung der Standardabwei-

chung verwendet werden. Ein Blattknoten ist ein Knoten, der keine Töchterknoten besitzt. Ein Blattknoten ist entweder ein konstanter Wert oder ein einfacher Unterabfrageknoten. Ein Unterabfrageknoten kann eine Anzahl an Komponenten-Unterabfragen besitzen. Jede Komponenten-Unterabfrage wird auch mit Hilfe eines entsprechenden Abfrageberechnungsbaums spezifiziert. Die Abfrageberechnungsbäume werden zum Abfrageprozessor **608** übertragen.

Der Abfrageprozessor **608** ist für die Verwaltung der Verarbeitung der Abfragen verantwortlich. Beim Empfang eines Abfrageberechnungsbaums vom Abfrage-Parser weist dieser der Abfrage einen Abfragebezeichner (QID) zu, und er weist auch jedem Blattknoten, der eine Unterabfrage spezifiziert, einen Unterabfragebezeichner (QSID) zu. Eine Unterabfrage, die keine Komponentenunterabfragen besitzt, wird als Einfache Unterabfrage bezeichnet. Eine Unterabfrage wird verarbeitet, indem eine Unterabfragemeldung mittels des Kommunikationsmoduls **604** zum spezifizierten Indexknoten übertragen wird. Der Abfrageprozessor **608** verarbeitet an dem spezifizierten Bestimmungsknoten die Unterabfragemeldung durch Übertragung einer Einfachen Unterabfragemeldung zum Fragmenttabellenmodul **602**, das mit einer Unterabfrageantwortmeldung reagiert. Der Abfrageprozessor **608** sendet danach die Unterabfrageantwortmeldung zum Indexknoten, der ursprünglich die Unterabfragemeldung geschickt hat. Als Ergebnis sendet und empfängt der Abfrageprozessor **608** Unterabfragemeldungen und Unterabfrageantwortmeldungen. Wenn Unterabfrageantwortmeldungen empfangen werden, wird die im Abfrageberechnungsbaum spezifizierte Verarbeitung durchgeführt. Wenn eine Unterabfrage eine Komponentenunterabfrage besitzt, fordert die Unterabfrage die Verarbeitung zusätzlicher Unterabfragen an. Wenn die gesamte Abfrage (einschließlich aller ihrer Unterabfragen und deren Unterabfragen, usw., die als "verschachtelte Unterabfragen" bezeichnet werden) berechnet wurde, werden die Ergebnisse formatiert und zum Front-End-Rechner übertragen, von dem die Abfrage erhalten wurde. Zum Beispiel können die Ergebnisse als Graph oder Tabelle angegeben werden. Da jede Abfrage oder verschachtelte Unterabfrage einer Ebene innerhalb des Baumes zugeordnet ist, ist der Abfrageprozessor **608** demgemäß für die Verarbeitung von Abfragen aller Ebenen innerhalb des Baumes verantwortlich.

Fig. 7 zeigt eine herkömmliche Systemarchitektur für ein beispielhaftes Computersystem **800**. Jeder der Anwendercomputer, Front-End-Computer und der Computer-Knoten einschließlich den Indizier- und Warehouse-Knoten kann als eine Instanz des Computersystems **800** implementiert werden. Das beispielhafte Computersystem von **Fig. 7** wird jedoch ausschließlich für beschreibende Zwecke diskutiert und sollte nicht als Einschränkung der Erfindung betrachtet werden. Wenngleich sich die folgende Beschreibung auf Begriffe beziehen kann, die allgemein zur Beschreibung bestimmter Computersysteme verwendet werden, gelten die beschriebenen Konzepte gleichermaßen auch für andere Computersysteme, einschließlich Systeme, deren Architektur jener unähnlich ist, die in **Fig. 7** dargestellt ist.

Das Computersystem **800** umfaßt eine zentrale Recheneinheit (CPU) **805**, die einen herkömmlichen Mikroprozessor enthalten kann, einen Direktzugriffsspeicher (RAM) **810** zum temporären Speichern von Informationen, und einen Nur-Lesen-Speicher (ROM) **815** für die dauerhafte Speicherung von Informationen. Eine Speichersteuerung **820** ist vorhanden, welche den System-RAM **810** steuert. Eine Bussteuerung **825** ist vorhanden, um den Bus **830** zu steuern, und eine Unterbrechungssteuerung **835** wird verwendet, um verschiedene Unterbrechungssignale von anderen Systemkomponenten zu empfangen und zu verarbeiten.

Ein Massenspeicher kann durch Diskette **842**, CD-ROM **847** oder Festplatte **852** zur Verfügung gestellt werden. Daten und Software können mit einem Client-Computer **800** über austauschbare Medien, wie zum Beispiel Diskette **842** und CD-ROM **847**, ausgetauscht werden. Die Diskette **842** kann in das Diskettenlaufwerk **841** eingeführt werden, welches durch den Controller **840** mit dem Bus **830** verbunden ist. Auf ähnliche Weise kann die CD-ROM **847** in das CD-ROM-Laufwerk **846** eingeführt werden, welches durch den Controller **845** mit dem Bus **830** verbunden ist. Schließlich ist die Festplatte **852** Teil eines Festplattenlaufwerks **851**, das vom Controller **850** mit dem Bus **830** verbunden wird.

Die Benutzereingabe in das Computersystem **800** kann durch eine Anzahl unterschiedlicher Geräte erfolgen. Zum Beispiel können eine Tastatur **856** und eine Maus **857** über einen Tastatur- und Maus-Controller **855** mit dem Bus **830** verbunden sein. Ein Audiowandler **896**, der sowohl als Mikrofon als auch als Lautsprecher dienen kann, wird vom Audio-Controller **897** mit dem Bus **830** verbunden. Für Fachleute dieses Bereiches sollte leicht erkennbar sein, daß auch andere Eingabevorrichtungen, wie zum Beispiel ein Stift und/oder ein Tablett, oder ein Mikrofon für die Spracheingabe, über den Bus **830** und einen entsprechenden Controller am Client-Computer **800** angeschlossen werden können. Ein DMA-Controller **860** ist vorhanden, um direkten Speicherzugriff auf den System-RAM **810** zu ermöglichen. Eine Sichtanzeige wird von einem Video-Controller **865** erzeugt, der den Monitor **870** steuert.

Das Computersystem **800** umfaßt auch einen Netzwerkadapter **890**, der es dem Client-Computer ermöglicht, über einen Bus **891** mit einem Netzwerk **895** verbunden zu werden. Das Netzwerk **895**, bei dem es sich um ein lokales Netzwerk (LAN), ein Weitverkehrsnetz (WAN) oder das Internet handeln kann, kann Allzweck-Kommunikationsleitungen verwenden, welche mehrere Netzwerkgeräte miteinander verbinden.

Das Computersystem **800** wird im allgemeinen von einer Betriebssystemsoftware gesteuert und koordiniert. Zusätzlich zu anderen Computersystemkontrollfunktionen steuert das Betriebssystem auch die Zuordnung der Systemressourcen und die Ausführung von Aufgaben, wie zum Beispiel die Rechenzeitvergabe, die Speicherverwaltung sowie die Netzwerk- und Eingabe-/Ausgabedienste.

Eine Software-Implementierung von Komponenten der oben beschriebenen Ausführungsform kann Computeranweisungen und Computerroutinen umfassen, die sich entweder auf einem berührbaren Medium, wie zum einem computerlesbaren Medium, z. B. der Diskette **842**, der CD-ROM **847**, im ROM **815**, oder der Festplatte **852** von **Fig. 7** befinden, oder die über ein Modem oder ein anderes Schnittstellengerät, wie zum Beispiel den Kommunikationsadapter **890**, der am Netzwerk **895** angeschlossen ist, über ein Medium **891** übertragen werden. Bei dem Medium **891** kann es sich entweder um ein berührbares Medium handeln, wobei es sich unter anderem auch um optische oder hartverdrahtete Kommunikationsleitungen handeln kann, oder welches mit drahtlosen Techniken implementiert sein kann, wozu unter anderem Mikrowellen-, Infrarot- oder andere Übertragungstechniken zählen. Es kann sich dabei auch um das Internet handeln. Bei einer derartigen Übertragung können die Softwarekomponenten die Form eines digitalen Signals annehmen, das sich in einer Trägerwelle befindet. Eine Reihe von Computerbefehlen enthält alle oder einige der Funktionen, die zuvor im Hinblick auf die Erfindung beschrieben wurden. Fachleute dieses Bereichs werden anerkennen, daß solche Computeranweisungen in einer Vielzahl von Programmiersprachen geschrieben und in zahlreichen Computerarchitekturen oder Betriebssystemen verwendet werden kön-

nen. Weiter können solche Anweisungen mit Hilfe jeder beliebigen zukünftigen oder gegenwärtigen Speichertechnologie gespeichert werden, wobei unter anderem Halbleiter-, magnetische, optische oder andere Speichervorrichtungen zu erwähnen sind, oder mit Hilfe einer beliebigen gegenwärtigen oder zukünftigen Kommunikationstechnologie übertragen werden können, wobei unter anderem optische, Infrarot-, Mikrowellen- oder andere Übertragungstechnologien zu erwähnen sind. Es wird auch überlegt, ein solches Computerprogramm als entfernbares Medium mit einer begleitenden gedruckten oder elektronischen Dokumentation zu verteilen, z. B. als in Schrumpfhülle verpackte Software, vorinstalliert auf einem Computersystem, z. B. auf einem System-ROM oder auf Festplatte, oder verteilt von einem Server oder einem elektronischen Mailboxsystem über ein Netzwerk, z. B. dem Internet oder dem World Wide Web.

Wenngleich eine beispielhafte Ausführungsform der Erfindung offenbart wurde, ist es für Fachleute dieses Bereiches leicht ersichtlich, daß verschiedene Änderungen und Modifizierungen durchgeführt werden können, welche einige der Vorteile der Erfindung mit sich bringen, ohne dadurch von Geist und Umfang der Erfindung abzuweichen. Für Fachleute dieses Bereiches ist es weiter offensichtlich, daß andere Komponenten, welche die selben Funktionen ausüben, anstelle der hier genannten verwendet werden können. Weiter können die Methoden der Erfindung entweder in ausschließlichen Softwareimplementationen mit Hilfe der entsprechenden Prozessoranweisungen oder in Hybridimplementationen umgesetzt werden, welche eine Kombination aus Hardware-Logik und Software-Logik verwenden, um die selben Ergebnisse zu erzielen. Weitere Aspekte, wie zum Beispiel die Speichergröße, die spezifische Konfiguration der zur Erzielung einer bestimmten Funktion notwendigen Logik und/oder Anweisungen, sowie andere Modifizierungen am erfinderischen Konzept sollen von den angehängten Ansprüchen abgedeckt werden. Daher sollte die Erfindung nur insofern als eingeschränkt betrachtet werden, als dies durch den Umfang der Ansprüche angezeigt wird.

Patentansprüche

1. Verfahren zur Warehouse-Verarbeitung von Objekten oder Positionen von Objekten auf eine Art und Weise, die der Wissensextraktion mit Hilfe von Abfragen in einem verteilten Computer-Datenbanksystem mit einer Vielzahl an Indexknoten und einer Vielzahl an Warehouse-Knoten dienlich ist, die über ein Netzwerk miteinander verbunden sind, wobei das Verfahren die folgenden Schritte umfaßt:

A) das Extrahieren einer ersten Anzahl an Merkmalen aus einem Objekt, das von einer anderen Datenbank heruntergeladen wurde, durch einen Warehouse-Knoten;

B) das Fragmentieren der einzelnen extrahierten Objektmerkmale in eine Anzahl von Objektmerkmalsfragmenten;

C) das Streuspeichern der einzelnen Objektmerkmalsfragmente der ersten Anzahl an Objektmerkmalen durch den Warehouse-Knoten, wobei jedes der streugespeicherten Objektmerkmalsfragmente einen ersten Abschnitt und einen zweiten Abschnitt besitzt;

D) das Übertragen der einzelnen streugespeicherten Objektmerkmalsfragmente der ersten Anzahl an Merkmalsfragmenten durch den Warehouse-Knoten zu einem jeweiligen der Vielzahl an Indexknoten, die vom ersten Abschnitt des jeweiligen streugespeicherten Objektmerkmals angegeben werden;

E) das Verwenden des zweiten Abschnitts des jeweiligen streugespeicherten Objektmerkmalsfragments durch den Indexknoten, um auf Daten gemäß einer lokalen Streuspeichertabelle, die sich am Indexknoten befindet, zuzugreifen;

F) das Zurückgeben einer Anzahl an Objekt-Bezeichnern, welche den Daten, auf die zugegriffen wurde, entsprechen, an den Warehouse-Knoten durch die einzelnen Indexknoten, welche auf Daten gemäß dem jeweiligen streugespeicherten Objektmerkmalsfragment zugreifen;

G) das Bestimmen durch den Warehouse-Knoten, ob das Objekt einem Objektbezeichner aus der Anzahl der Objektbezeichner zuzuordnen ist, oder ob das Objekt einem Objektbezeichner zuzuordnen ist, der noch nicht in Verwendung steht;

H) das Zuordnen eines Objektbezeichners zum Objekt gemäß der Bestimmung durch den Warehouse-Knoten;

I) das Extrahieren einer zweiten Anzahl an Merkmalen aus dem Objekt durch den Warehouse-Knoten;

J) das Fragmentieren der extrahierten zweiten Anzahl an Objektmerkmalen in eine Anzahl an Objektmerkmalsfragmenten;

K) das Streuspeichern der einzelnen Objektmerkmalsfragmente der zweiten Anzahl an Objektmerkmalen durch den Warehouse-Knoten, wobei das streugespeicherte Objektmerkmalsfragment einen ersten Abschnitt und einen zweiten Abschnitt besitzt;

L) das Übertragen der einzelnen streugespeicherten Objektmerkmalsfragmente der zweiten Anzahl an Merkmalsfragmenten durch den Warehouse-Knoten zu einem jeweiligen der Vielzahl an Indexknoten, die vom ersten Abschnitt des jeweiligen streugespeicherten Objektmerkmalfragments angegeben werden; und

M) das Verwenden des zweiten Abschnitts des jeweiligen streugespeicherten Objektmerkmalsfragments durch den Indexknoten, um Daten gemäß einer lokalen Streuspeichertabelle, die sich am Indexknoten befindet, zu speichern.

2. Verfahren nach Anspruch 1, weiter umfassend den Schritt des Bestimmens eines Maßes der Ähnlichkeit durch den Warehouse-Knoten zwischen den Daten, auf die zugegriffen wird, und dem Objekt, nachdem der Schritt der Rückgabe der ersten Anzahl an Objektbezeichnern ausgeführt wurde.

3. Verfahren nach Anspruch 2, wobei das Maß der Ähnlichkeit bestimmt wird durch eine Ähnlichkeitsfunktion, die auf Merkmalen basiert, welche sowohl den Daten, auf die zugegriffen wird, als auch dem Objekt eigen sind, und Merkmalen, die nur dem Objekt eigen sind.

4. Verfahren zur Durchführung von Data Mining-Aktivitäten unter Verwendung von Abfragen in einem verteilten Computer-Datenbanksystem mit einer Anzahl an Indexknoten, die mit einem Netzwerk verbunden sind, wobei das Verfahren die folgenden Schritte umfaßt:

A) das Auswählen eines ersten aus der Anzahl an Indexknoten, der im folgenden als Heimknoten der Abfrage bezeichnet wird;

B) das Extrahieren einer Anzahl an Unterabfragen durch den Heimknoten aus einer Abfrage durch einen Anwender, wobei jede Unterabfrage

ein Merkmal, eine Anzahl von Unterabfragen und eine Berechnungsspezifizierung umfaßt;

C) das Fragmentieren der einzelnen Unterabfragemerkmale in eine Anzahl an Unterabfragemerkmalsfragmente;

D) das Streuspeichern durch den Heimknoten der einzelnen Unterabfragemerkmalsfragmente der einzelnen Unterabfragemerkmalsfragmente, wobei jedes der streugespeicherten Unterabfragemerkmalsfragmente einen ersten Abschnitt und einen zweiten Abschnitt besitzt;

E) das Übertragen der einzelnen streugespeicherten Unterabfragemerkmalsfragmente durch den Heimknoten zu einem jeweiligen der Anzahl an Indexknoten, die vom ersten Abschnitt der einzelnen streugespeicherten Unterabfragemerkmalsfragmente angegeben werden;

F) das Verwenden des zweiten Abschnitts des jeweiligen streugespeicherten Unterabfragemerkmalsfragments durch den Indexknoten, um auf Daten gemäß einer lokalen Streuspeichertabelle, die sich am Indexknoten befindet, zuzugreifen;

G) das rekursive Evaluieren der einzelnen Unterabfragen der Anzahl an Unterabfragen, die in der jeweiligen Unterabfrage enthalten sind, welche vom Heimknoten übertragen wurde, durch den Indexknoten, wobei der Indexknoten als Heimknoten der Unterabfrage der Anzahl an Unterabfragen fungiert;

H) das Berechnen der Informationen gemäß der Berechnungsspezifikation der jeweiligen vom Heimknoten übertragenen Unterabfrage durch den Indexknoten gemäß den Daten, auf die zugegriffen wurde, und den Informationen, die durch die rekursive Evaluierung der einzelnen Unterabfragen der Anzahl an Unterabfragen bestimmt werden, die in der jeweiligen vom Heimknoten übertragenen Unterabfrage enthalten sind;

I) das Zurückgeben der Informationen zum Heimknoten durch den jeweiligen Indexknoten.

5. Verfahren nach Anspruch 4, weiter umfassend den Schritt des Empfangens der Abfrage vom Anwender am Heimknoten vor dem Schritt des Extrahierens von Unterabfragen aus der Abfrage.

6. Verteiltes Computer-Datenbanksystem für das Warehousing von Informationsobjekten oder Positionen von Informationsobjekten, umfassend:

A) eine Anzahl an Warehouse-Knoten (**108**) und eine Anzahl an Indexknoten (**106**), wobei die Anzahl an Warehouse-Knoten und die Anzahl an Indexknoten durch ein Netzwerk (**110**) miteinander verbunden sind;

B) wobei jeder Warehouse-Knoten (**108**) beim Herunterladen eines Objekts eine erste Anzahl an Merkmalen aus dem Objekt extrahiert, jedes der Objektmerkmale in ein Objektmerkmalfragment fragmentiert, jedes der Objektmerkmalsfragmente in ein streugespeichertes Objektmerkmalsfragment mit einem ersten Abschnitt und einem zweiten Abschnitt streuspeichert, und jedes der streugespeicherten Objektmerkmalsfragmente zu einem jeweiligen der Anzahl an Indexknoten (**106**) überträgt, der vom ersten Abschnitt des streugespeicherten Objektmerkmalsfragments angegeben wird;

C) wobei jeder der Indexknoten (**106**) den zweiten Abschnitt des streugespeicherten Abfragemerkmalfragments verwendet, um auf die Daten gemäß einer lokalen Streuspeichertabelle zuzugreifen, die sich am Indexknoten befindet, und eine Vielzahl an Objekt-Bezeichnern, welche den Daten, auf die zugegriffen wurde, entsprechen, an den Warehouse-Knoten (**108**) zurückgibt;

D) wobei das Warehouse dem Objekt entweder einen der Objektbezeichner der Anzahl an Objektbezeichnern oder einen noch nicht verwendeten Objektbezeichner zuordnet, eine zweite Anzahl an Merkmalen aus dem Objekt extrahiert, jedes der extrahierten Merkmale der zweiten Anzahl an Fragmenten in eine Anzahl an Objektmerkmalsfragmenten extrahiert; jedes der Objektmerkmalsfragmente der zweiten Anzahl an Objektmerkmalen in ein streugespeichertes Objektmerkmal mit einem ersten und zweiten Abschnitt streuspeichert, und jedes der streugespeicherten Objektmerkmalsfragmente zu einem jeweiligen der Anzahl an Indexknoten überträgt, der vom ersten Abschnitt des streugespeicherten Objektmerkmalfragments angegeben wird;

E) wobei jeder Indexknoten den zweiten Abschnitt des jeweiligen streugespeicherten Objektmerkmalsfragments verwendet, um die Objekte oder Positionen der Objekte gemäß einer lokalen Streuspeichertabelle, die sich am Indexknoten befindet, zu speichern.

7. Verteiltes Computer-Datenbanksystem nach Anspruch 6, wobei der Warehouse-Knoten ein Maß der Ähnlichkeit zwischen den Daten, auf die zugegriffen wird, und dem Objekt bestimmt, um dem Objekt einen Objektbezeichner zuzuordnen.

8. Verfahren nach Anspruch 7, wobei der Warehouse-Knoten die Ähnlichkeit mit Hilfe einer Ähnlichkeitsfunktion mißt, die von Merkmalen bestimmt wird, welche sowohl den Daten, auf die zugegriffen wird, als auch dem Objekt eigen sind; und Merkmalen, die nur dem Objekt eigen sind.

9. Verteiltes Computer-Datenbanksystem mit einem Data Mining-Werkzeug für die Handhabung von Abfragen von einem Anwender, umfassend:

A) eine Anzahl Indexknoten (**106**), die über ein Netzwerk (**110**) miteinander verbunden sind;

B) wobei jeder der Indexknoten bei Empfang einer Abfrage von einem Anwender, woraufhin er als Heimknoten der Abfrage bezeichnet wird, eine Anzahl an Unterabfragen von der Abfrage und eine Anzahl an Merkmalen von jeder Unterabfrage extrahiert, jedes der Unterabfragemerkmale in eine Anzahl an Unterabfragemerkmalsfragmente fragmentiert, das Unterabfragemerkmal der Anzahl an Unterabfragen in ein streugespeichertes Unterabfragemerkmal mit einem ersten und einem zweiten Abschnitt streuspeichert, und jedes der streugespeicherten Unterabfragemerkmalsfragmente zu einem jeweiligen der Anzahl an Indexknoten überträgt, der vom ersten Abschnitt des streugespeicherten Unterabfragemerkmalsfragments angegeben wird;

C) wobei weiter jeder der Indexknoten den zweiten Abschnitt des streugespeicherten Unterabfragemerkmalsfragments verwendet, um auf Daten gemäß einer lokalen Streuspeichertabelle zurückzugreifen, die sich am Indexknoten befindet, jede Unterabfrage, die in der jeweiligen Unterabfrage enthalten ist, rekursiv evaluiert, die Informationen gemäß den Daten, auf die zugegriffen wird, und den Informationen, die von der rekursiven Evalu-

ierung bestimmt werden, berechnet, und die Informationen an den Heimknoten zurückgibt.

10. Verteiltes Computer-Datenbanksystem für das Warehousing und Data Mining, umfassend:

A) eine Anzahl Warehouse-Knoten (**108**) und eine Anzahl an Indexknoten (**106**), wobei die Anzahl an Warehouse-Knoten und die Anzahl an Indexknoten durch ein Netzwerk miteinander verbunden sind;

B) wobei jeder der Warehouse-Knoten bei Empfang eines Download-Befehls eine vorherbestimmte Aufgabe als Reaktion auf den Download-Befehl in eine Warteschlange stellt;

C) eine als Reaktion auf einen Download-Befehl in eine Warteschlange gestellte Download-Aufgabe eine erste Anzahl an Merkmalen aus einem Objekt extrahiert, das vom Download-Befehl heruntergeladen wurde, jedes der Objektmerkmale in eine Anzahl an Objektmerkmalsfragmenten fragmentiert, jedes der Objektmerkmalsfragmente der ersten Anzahl an Objektmerkmalen in ein streugespeichertes Objektmerkmalsfragment mit einem ersten Abschnitt und einem zweiten Abschnitt streuspeichert, und eine Antwortmeldung, welche jedes der streugespeicherten Objektmerkmalsfragmente enthält, zu einem jeweiligen der Anzahl an Indexknoten überträgt, der vom ersten Abschnitt des streugespeicherten Objektmerkmalsfragments angegeben wird;

D) wobei der Indexknoten bei Empfang der geholten Meldung den zweiten Abschnitts des streugespeicherten Objektmerkmalsfragments verwendet, um auf die Daten gemäß einer lokalen Streuspeichertabelle zuzugreifen, die sich am Indexknoten befindet, und eine Meldung an den Warehouse-Knoten überträgt, worin er eine Anzahl an Objektbezeichnern entsprechend den Daten, auf die zugegriffen wurde, zurückgibt;

E) wobei der Warehouse-Knoten bei Empfang der Anzahl an Objektbezeichnern von der Anzahl an Indexknoten dem Objekt entweder einen der Objektbezeichner aus der Anzahl an Objektbezeichnern oder einen noch nicht verwendeten Objektbezeichner zuordnet, eine zweite Anzahl an Merkmalen aus dem Objekt extrahiert, jedes der Objektmerkmale der zweiten Anzahl an Objektmerkmalen in eine Anzahl an Objektmerkmalsfragmenten fragmentiert; jedes der Objektmerkmalsfragmente der zweiten Anzahl an Objektmerkmalsfragmenten in ein streugespeichertes Objektmerkmalsfragment mit einem ersten und einem zweiten Abschnitt fragmentiert, und eine Einfügemeldung, die jedes der streugespeicherten Objektmerkmalsfragmente enthält, zu einem jeweiligen der Anzahl an Indexknoten überträgt, der vom ersten Abschnitt des streugespeicherten Objektmerkmalsfragments angegeben wird;

F) wobei der Indexknoten bei Empfang der Einfügemeldung den zweiten Abschnitt des streugespeicherten Objektmerkmalsfragments verwendet, um Daten gemäß einer lokalen Streuspeichertabelle, die sich am Indexknoten befindet, zu speichern.

11. Verteiltes Computer-Datenbanksystem nach Anspruch 10, wobei der Warehouse-Knoten (**108**) in Maß der Ähnlichkeit zwischen den Daten, auf die zugegriffen wird, und dem Objekt bestimmt, um dem Objekt einen Objektbezeichner zuzuordnen.

12. Verfahren nach Anspruch 11, wobei der Warehouse-Knoten die Ähnlichkeit mit Hilfe einer Ähnlichkeitsfunktion mißt, die bestimmt wird durch: Merkmale, die sowohl den Daten, auf die zugegriffen wird, als auch dem Objekt eigen sind, und Merkmalen, die nur dem Objekt eigen sind.

13. Verteiltes Computer-Datenbanksystem mit einem Data Mining-Werkzeug für die Handhabung von Abfragen von einem Anwender, umfassend:

A) eine Anzahl an Indexknoten (**106**), die über ein Netzwerk (**108**)miteinander verbunden sind;

B) wobei jeder Indexknoten bei Empfang eines Befehls von einem Anwender – wobei dieser Indexknoten als Heimknoten (**107**) des Befehls bezeichnet wird – eine vorherbestimmte Aufgabe als Reaktion auf den Befehl in eine Warteschlange stellt;

C) eine in eine Warteschlange gestellte Abfrageaufgabe dazu führt, daß als Reaktion auf einen Abfragebefehl vom Anwender eine Anzahl an Unterabfragen von einer Abfrage, die im Abfragebefehl enthalten ist, und eine Anzahl an Merkmalen aus jeder der extrahierten Unterabfragen extrahiert wird, jedes der Unterabfragemerkmale in eine Anzahl an Unterabfragemerkmalsfragmente fragmentiert wird, jedes der Unterabfragemerkmalsfragmente in ein streugespeichertes Unterabfragefragment mit einem ersten Abschnitt und einem zweiten Abschnitt streugespeichert wird, und eine Unterabfragemeldung, welche jedes der streugespeicherten Unterabfragefragmente enthält, zu einem jeweiligen der Anzahl an Indexknoten übertragen wird, der vom ersten Abschnitt des streugespeicherten Unterabfragemerkmalsfragments angegeben wird;

D) wobei der Indexknoten bei Empfang der Unterabfragemeldung den zweiten Abschnitt des streugespeicherten Unterabfragemerkmalsfragments verwendet, um auf Daten gemäß einer lokalen Streuspeichertabelle zuzugreifen, die im Indexknoten vorhanden ist, jede Unterabfrage, die in der jeweiligen Unterabfrage vorhanden ist, rekursiv evaluiert, die Informationen gemäß den Daten, auf die zugegriffen wurde, und den Informationen, die durch die rekursive Evaluierung bestimmt wurden, berechnet, und eine Meldung überträgt, in der die Informationen zum Heimknoten zurückgegeben werden.

14. Verfahren nach Anspruch 13, wobei die Abfragemeldung vorherbestimmte Daten vom Indexknoten als Reaktion auf eine Abfrage anfordert, die im Abfragebefehl des Anwenders enthalten ist.

15. Informationswiedergewinnungsvorrichtung für die Verarbeitung einer Abfrage zur Wiedergewinnung von Informationen aus einer Datenbank, umfassend:

A) einen Mechanismus zur Auffindung einer Anzahl an Merkmalen und Merkmalsfragmenten in einem Index;

B) einen Evaluierungsmechanismus, gekoppelt mit dem Auffindungsmechanismus, der eine Anzahl an Unterabfragen einer Anzahl an Ebenen identifiziert, die in der Abfrage enthalten sind, und die Unterabfragen mit Hilfe der gefundenen Merkmale und Merkmalsfragmente evaluiert; und

C) einen Mechanismus, gekoppelt mit dem Evaluierungsmechanismus zum Sammeln und Speichern in einem Speicher einer Anzahl der Ergebnisse der rekursiven Evaluierung der Abfrage und

der Unterabfragen gemäß der Berechnung eines Gesamtergebnisses der Abfrage.

16. Verfahren für die Verarbeitung einer Abfrage zur Wiedergewinnung von Informationen aus einer Datenbank, umfassend:

A) das Auffinden einer Anzahl an Merkmalen und Merkmalsfragmenten in einem Index;

B) das Identifizieren einer Anzahl an Unterabfragen einer Anzahl an Ebenen, die in der Abfrage enthalten sind, und das rekursive Evaluieren der Unterabfragen mit Hilfe der gefundenen Merkmale und Merkmalsfragmente; und

C) das Sammeln und Speichern einer Anzahl an Ergebnissen der rekursiven Evaluierung der Abfrage und der Unterabfragen nach der Berechnung eines Gesamtergebnisses der Abfrage.

17. Computerprogramm-Produkt für die Verarbeitung einer Abfrage für die Wiedergewinnung von Informationen aus einer Datenbank, wobei das Computerprogramm-Produkt ein vom Computer ausführbares Programm umfaßt, welches sich auf einem vom Computer lesbaren Medium befindet, wobei das vom Computer ausführbare Programm umfaßt:

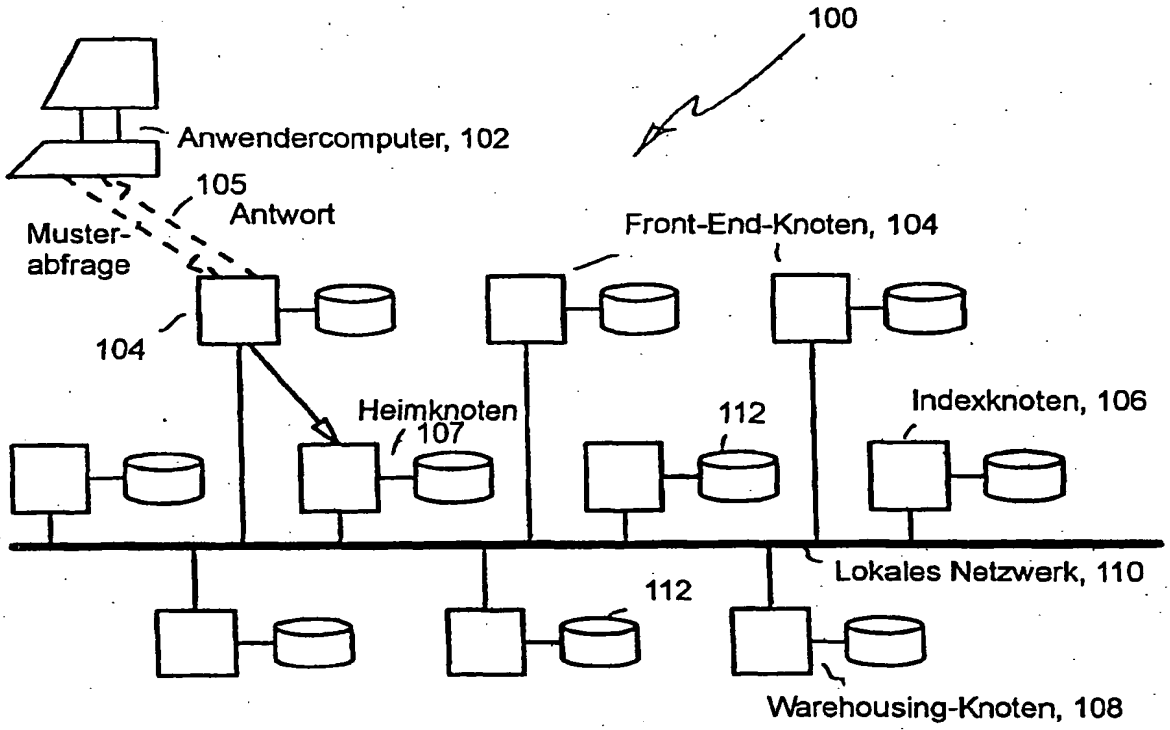
A) einen ersten Codeabschnitt zur Auffindung einer Anzahl an Merkmalen und Merkmalsfragmenten in einem Index;

B) einen zweiten Codeabschnitt zur Identifizierung einer Anzahl an Unterabfragen einer Anzahl an Ebenen, die in der Abfrage enthalten sind, und zum rekursiven Evaluieren der Unterabfragen mit Hilfe der gefundenen Merkmale und Merkmalsfragmente; und

C) einen dritten Codeabschnitt zum Sammeln und Speichern einer Anzahl an Ergebnissen der rekursiven Evaluierung der Abfrage und der Unterabfragen nach der Berechnung eines Gesamtergebnisses der Abfrage.

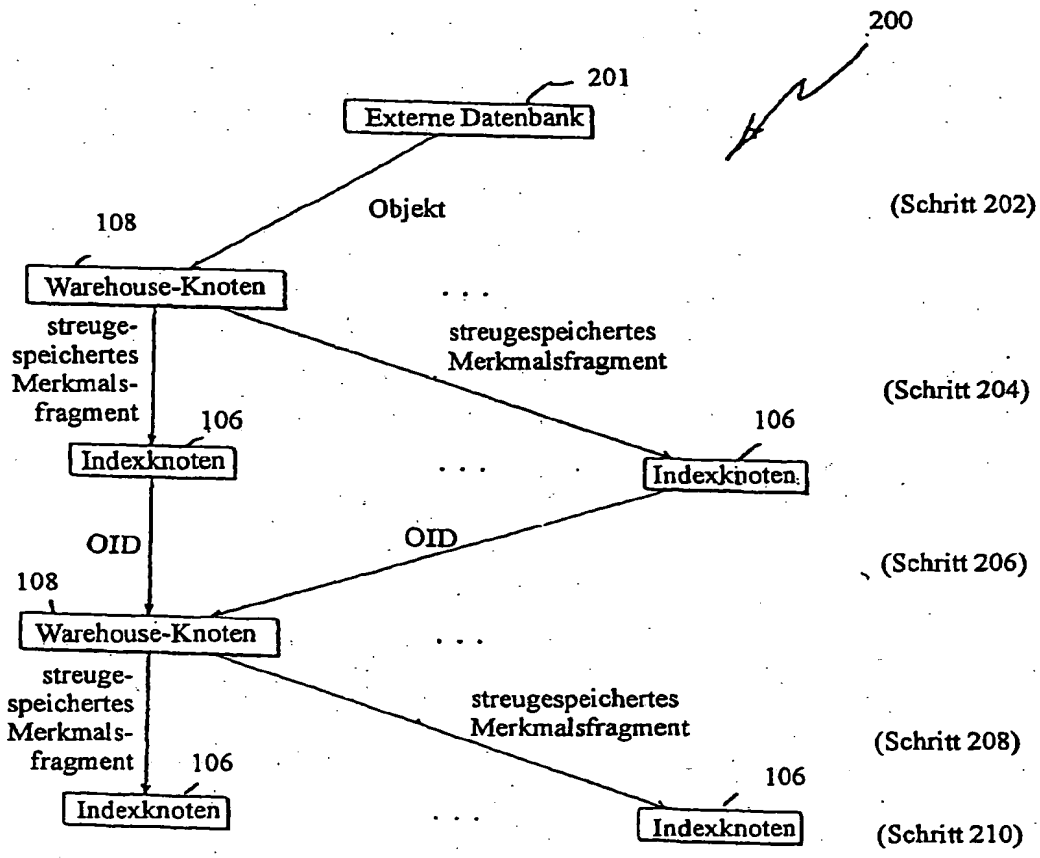
Hierzu 7 Seite(n) Zeichnungen

Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001



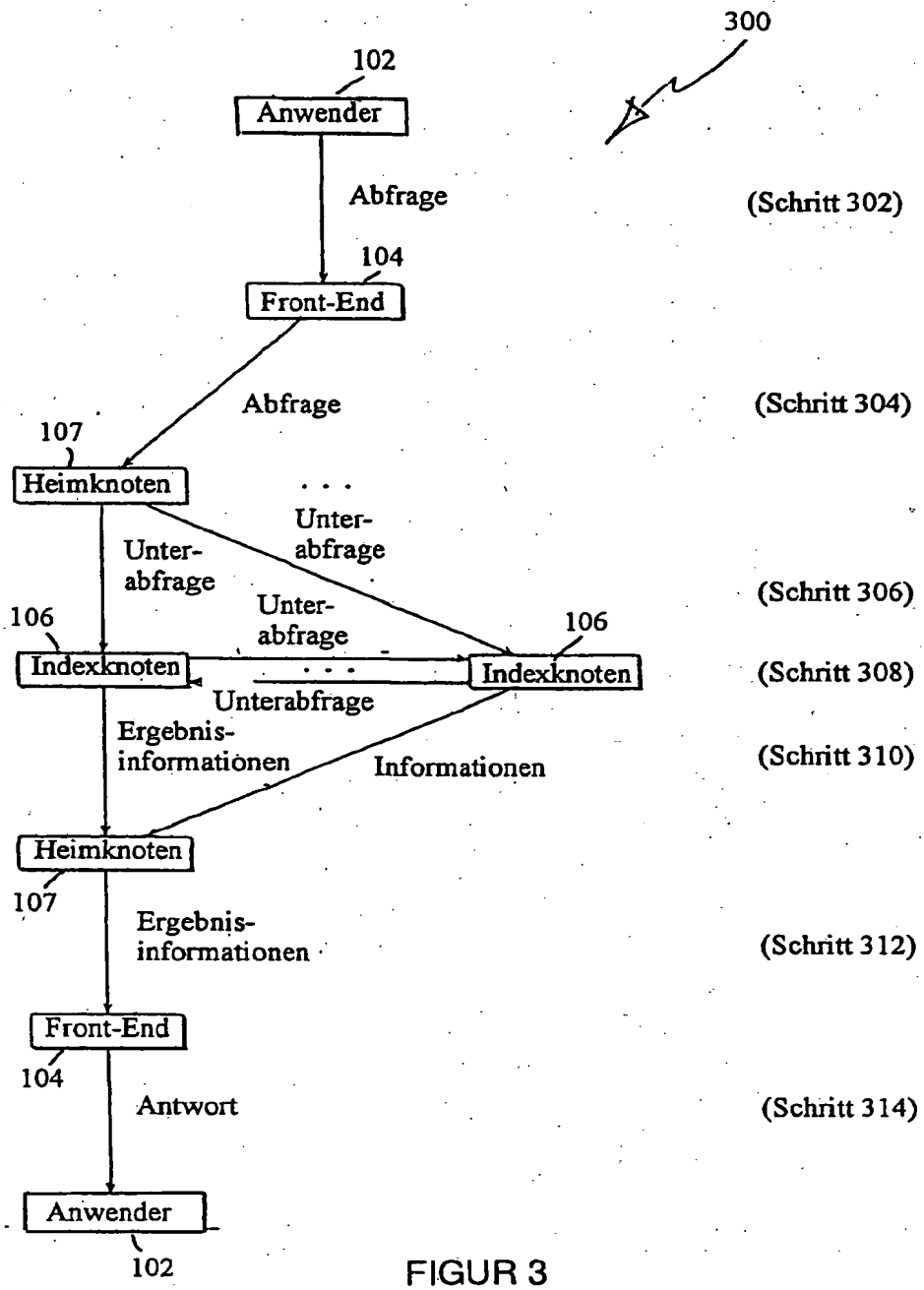
FIGUR 1

Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001



FIGUR 2

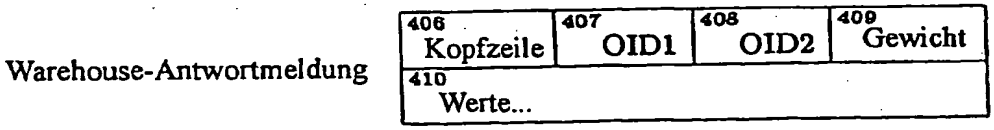
Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001



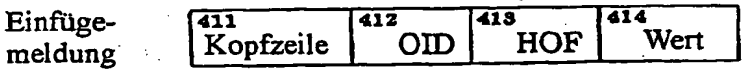
FIGUR 3

Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001

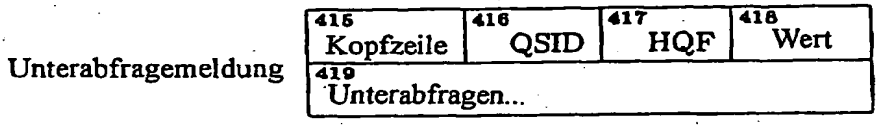
FIGUR 4a



FIGUR 4b



FIGUR 4c

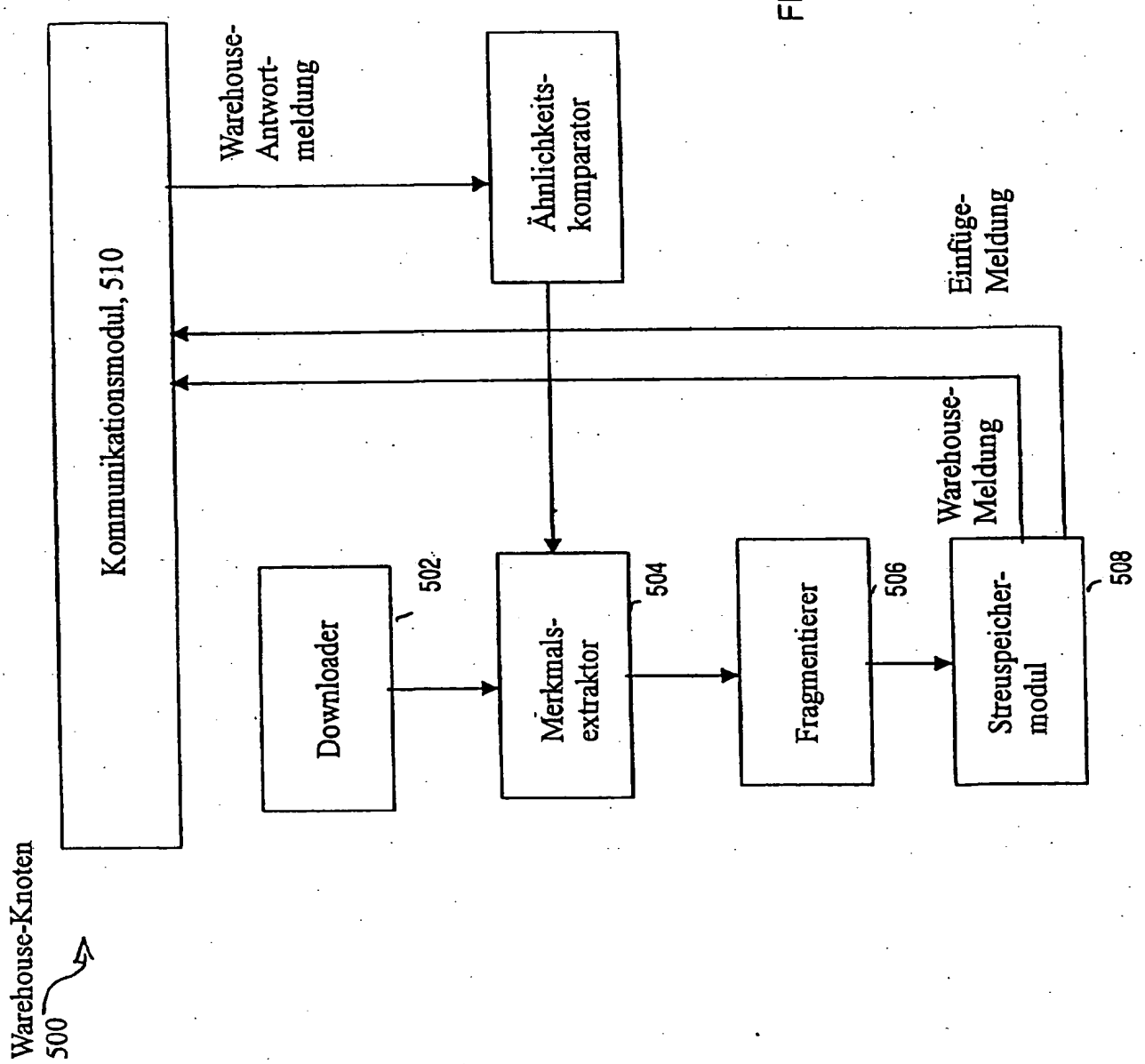


FIGUR 4d

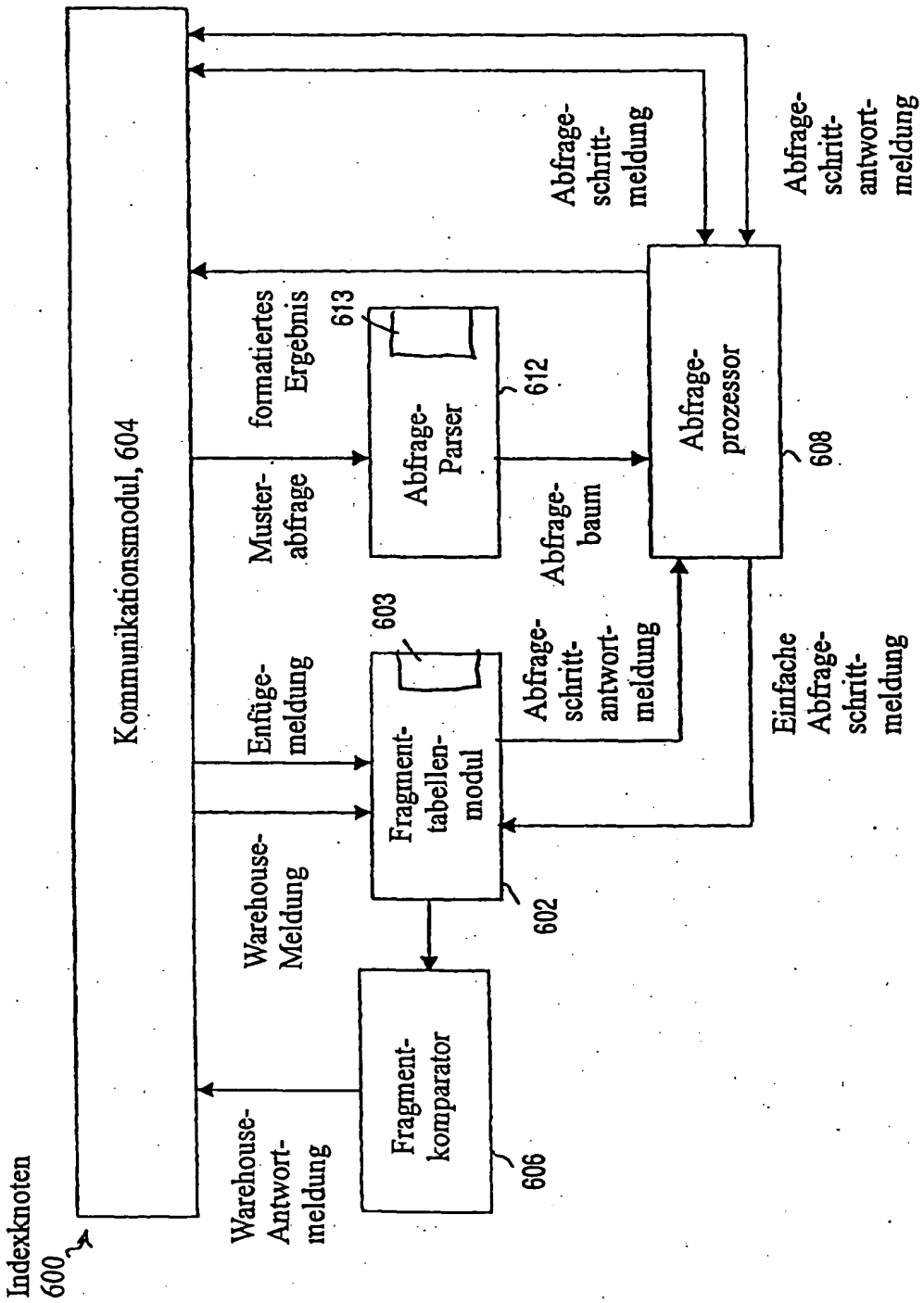
FIGUR 4e

Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001

FIGUR 5

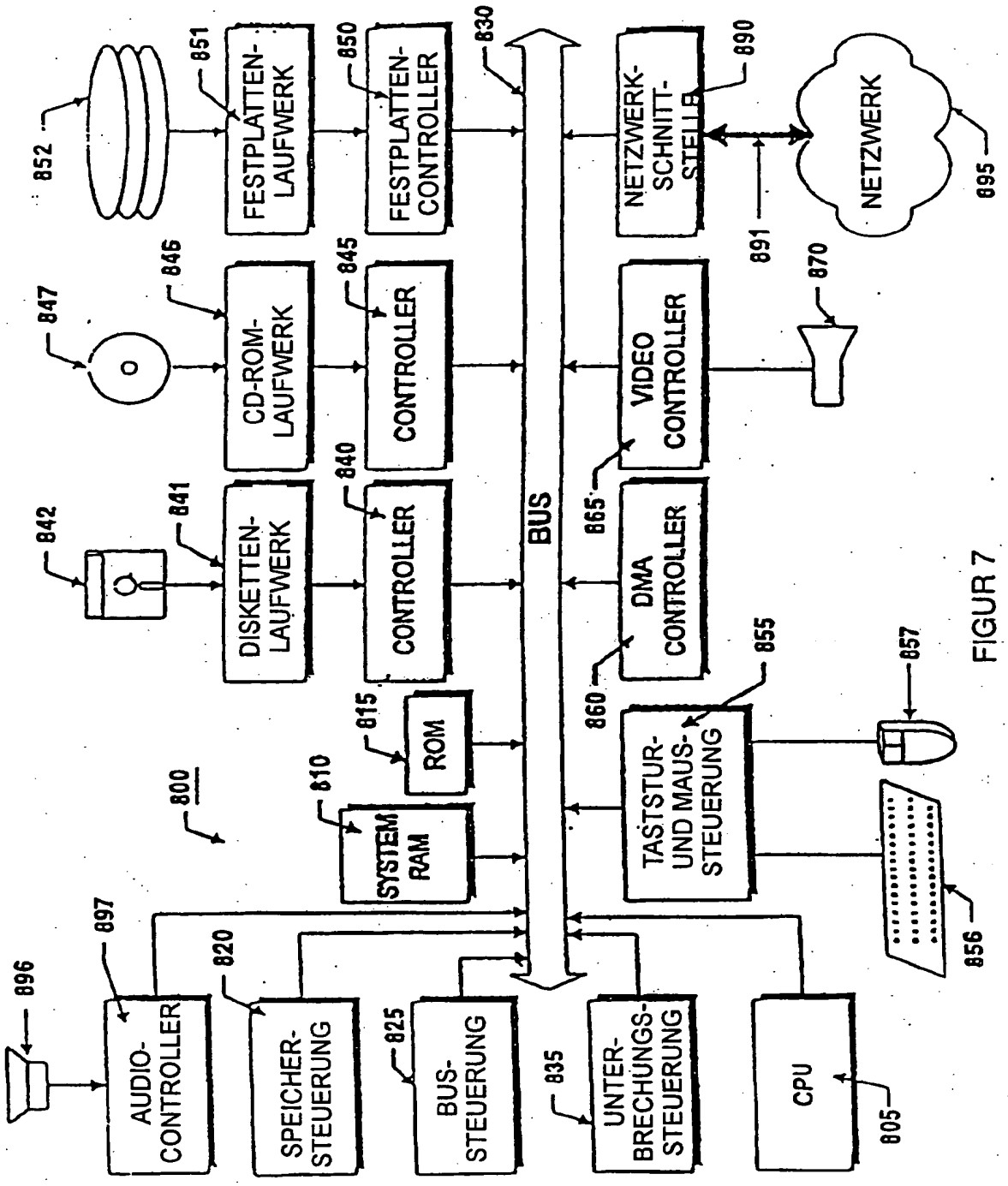


Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001



FIGUR 6

Nummer: DE 100 14 757 A1
Int. Cl.[7]: G 06 F 17/30
Offenlegungstag: 27. September 2001



FIGUR 7